Learning Curves for Gaussian Process regression on Random Graphs

Urry, Matthew; Urry, Matthew

Awarding institution:
King's College London

Thesis submitted to King's College London for the degree of Doctor of Philosophy

LEARNING CURVES FOR GAUSSIAN PROCESS REGRESSION ON RANDOM GRAPHS

April 30, 2013

Candidate: Matthew J. Urry

Supervisor: Prof. Peter Sollich

To Rebecca

ABSTRACT

Gaussian processes are a non-parametric method that can be used to learn both regression and classification rules from examples for arbitrary input spaces using the 'kernel trick'. They are well understood for inputs from Euclidean spaces, however, much less research has focused on other spaces. In this thesis I aim to at least partially resolve this. In particular I focus on the case where inputs are defined on the vertices of a graph and the task is to learn a function defined on the vertices from noisy examples, i.e. a regression problem.

A challenging problem in the area of non-parametric learning is to predict the generalisation error as a function of the number of examples or learning curve. I show that, unlike in the Euclidean case where predictions are either quantitatively accurate for a few specific cases or only qualitatively accurate for a broader range of situations, I am able to derive accurate learning curves for Gaussian processes on graphs for a wide range of input spaces given by ensembles of random graphs. I focus on the random walk kernel but my results generalise to any kernel that can be written as a truncated sum of powers of the normalised graph Laplacian.

I begin first with a discussion of the properties of the random walk kernel, which can be viewed as an approximation of the ubiquitous squared exponential kernel in continuous spaces. I show that compared to the squared exponential kernel, the random walk kernel has some surprising properties which includes a non-trivial limiting form for some types of graphs. After investigating the limiting form of the kernel I then study its use as a prior. I show that unlike translationally invariant kernels such as the squared

exponential kernel defined in continuous spaces, the more commonly found method of the normalising the kernel, by setting the prior scale of the functions it predicts by normalising by the average unnormalised prior variance, results in large discrepancies between the prior scale of a function amongst the vertices of the graph, a probabilistically unlikely scenario. I propose a solution to this in the form of a local normalisation, where the prior scale at each vertex is normalised locally as desired. To drive home the point about kernel normalisation I then examine the differences between the two kernels when they are used as a Gaussian process prior over functions defined on the vertices of a graph. I show using numerical simulations that the locally normalised kernel leads to a probabilistically more plausible Gaussian process prior.

After investigating the properties of the random walk kernel I then discuss the learning curves of a Gaussian process with a random walk kernel for both kernel normalisations in a matched scenario (where student and teacher are both Gaussian processes with matching hyperparameters). I show that by using the cavity method I can derive accurate predictions along the whole length of the learning curve that dramatically improves upon previously derived approximations for continuous spaces suitably extended to the discrete graph case.

The derivation of the learning curve for the locally normalised kernel required an additional approximation in the resulting cavity equations. I subsequently, therefore, investigate this approximation in more detail using the replica method. I show that the locally normalised kernel leads to a highly non-trivial replica calculation, that eventually shows that the approximation used in the cavity analysis amounts to ignoring some consistency requirements between incoming cavity distributions.

Finally I investigate the case of calculating the learning curves for a Gaussian process in a mismatched scenario. I focus in particular on a teacher distribution that is given by a Gaussian process with a random walk kernel but different hyperparameters. I show that in this case, by applying the cavity method, I am able once more to calculate accurate predictions of the learning curve. The resulting equations resemble the matched case over an inflated number of variables. To finish this thesis I examine the learning curves for varying degrees of model mismatch.

ACKNOWLEDGEMENTS

I WOULD LIKE to thank everyone who has helped me over the past few years whilst I have been studying for my Ph.D. the continued support from family, friends and colleagues has been invaluable throughout the course of my studies. First and foremost of the people to whom I explicitly wish to express my gratitude is my supervisor Prof. Peter Sollich for both his continued help throughout out my Ph.D. and his unceasing patience when I became stuck. My thanks also go to the rest of the Disordered Systems group and in particular Francesco, Tanguy, Katy, Akram, Mehmet and Marianne who made some of the more stressful and long days bearable. Finally I wish to thank my long suffering fiancée Rebecca for her understanding during all those occasions where I came home and worked late nights and weekends.

CONTENTS

List of Figures **9**

List of Tables **11**

1 Introduction **12**

2 Theoretical Background **17**

2.1 Gaussian Processes 18

2.1.1 From linear regression to Gaussian process regression 19

2.1.2 Kernels 25

2.1.3 Other methods for learning on graphs 33

2.1.4 Characterising the performance of Gaussian processes 35

2.2 Replica Method 42

2.2.1 The replica method for learning curves 43

2.3 Cavity method and belief propagation – Two sides of the same coin . . . 50

2.4 Random Graphs 55

2.4.1 Erdős-Rényi random graphs 56

2.4.2 Generalised random graphs 58

2.4.3 d-Regular Random Graphs 60

2.4.4 Arbitrary degree distributions 62

3 The random walk kernel as a Gaussian process prior **65**

3.1 Insight into the random walk kernel 65

3.1.1 Approaching the fully correlated kernel 67

3.2 Using the random walk kernel as a Gaussian process prior 70

3.3 The effect of normalisation on learning 74

4 Approximating the learning curve – the cavity method **81**

4.1 The graph specific generalisation error 81

4.2 Predicting the learning curve 83

4.2.1 Creating the generating partition function 83

4.2.2 Global normalisation 85

4.2.3 Local normalisation 96

4.3 Learning curves for large p 102

4.4 Summary 106

5 Understanding counterflow – The replica method **108**

5.1 Global normalisation 109

5.1.1 Deriving the 'effective single site' formulation 111

5.1.2 Approximating for large V 113

5.2 Local normalisation 118

5.2.1 Deriving the 'effective single site' formulation 120

5.2.2 Approximating for large V 121

5.2.3 Approximating for large L 123

5.3 Summary 133

6 Predicting the generalisation error with model mismatch **136**

6.1 The generalisation error with model mismatch 137

6.2 Creating the generating partition function 138

6.3 Deriving the graphical model form 140
6.4 Calculating the generalisation error 144
6.5 Summary 153

7 Concluding Remarks **155**
7.1 Summary of Results 155
7.2 Further Work 162

Appendices **169**

Appendix A Graph coverings and the random walk kernel **170**
A.1 Graph coverings 170
A.2 Application to the random walk kernel 174
A.3 Scaling for large p 177

Appendix B Replica Derivations **179**
B.1 The saddle point method a.k.a. the method of steepest descents 179
B.1.1 Laplace's Method 180
B.1.2 Saddle-point method 182
B.2 Graph Normaliser 186
B.3 Large L saddle point approximation for ϵ_g 188

Appendix C Mismatch Cavity Derivations **190**
C.1 Integration of (6.3.7) 190
C.2 Deriving the final graphical model form 191
C.3 Dealing with zero examples 193

References **195**

LIST OF FIGURES

2.1 Bayesian linear regression 21

2.2 Samples from GP priors 23

2.3 Samples from the random walk kernel 33

2.4 Diagram of the PAC-Bayes classifier for GP regression 39

2.5 Comparison of the EA and VB models of a spin glass 51

2.6 Diagram of the cavity method 53

2.7 Diagram of message updates and marginals for the cavity method 54

2.8 Erdős-Rényi random graphs 58

2.9 Power law random graphs 62

2.10 Regular random graphs 64

3.1 Iterated random walk kernel on a tree 68

3.2 Comparison of the kernel on regular trees and graphs 70

3.3 Gaussian process priors with global kernel normalisation 73

3.4 Gaussian process priors with the unaltered normalised Laplacian 74

3.5 Learning curves for locally and globally normalised kernels 76

3.6 Normalisation mismatch learning curves 78

3.7 Error variances for locally and globally normalised kernels 79

4.1 Cavity/Replica learning curves for global normalisation 92

4.2 Normalised histograms of the top left entries of cavity covariances 95

4.3 Cavity predictions of Gaussian process priors with global normalisation . 97

4.4 Cavity/Replica learning curves for local normalisation 101

4.5 Scaling of the learning curves for large p 105

5.1 Diagram to explain counterflow 130

5.2 Local posterior variances of GPs with the locally normalised kernel . . . 133

6.1 Mismatch learning curves 151

6.2 Mismatched learning curves for varied p_g 152

7.1 Community random graph 164

7.2 Community graph learning curves 166

A.1 Bethe lattice covering of the weighted Path 175

A.2 Learning curve scaling for large p 178

B.1 Cauchy's Theorem 183

B.2 The saddle point deformed path 184

C.1 Integration of a Gaussian with a linear constraint 192

LIST OF TABLES

2.1 Euclidean Kernels 30

2.2 Graph Kernels 32

CHAPTER 1

INTRODUCTION

Gaussian processes (GPs) have become an ubiquitous workhorse for probabilistic inference and have been invented many times under various guises. In geostatistics under the name *kriging* they were created to predict terrain between measurements [73], in discrete time control processes they were created in the form of *Kalman filters* and were applied for trajectory estimation in the Apollo missions [55] and finally in machine learning they have been created as a logical extension to *radial basis functions* (RBF) [80]. They owe their widespread use and reinvention to two important factors: firstly they are, at least in the basic setting, easy to use and understand, and secondly, they can incorporate prior beliefs about the problem in an intuitive way. In short they're transparent to the user and can be used with little knowledge of the underlying workings of GPs.

Although there is a large body of research into GPs (see for example Rasmussen and Williams [96] and references therein) due to the breadth of uses, thanks mainly to the 'kernel trick', there are many areas as of yet untouched. I aim in this thesis to make a small but significant dent in one of these unexplored areas: the approximation of the generalisation error of GPs for regression on large graphs.

In a nutshell the problem that GP regression on graphs addresses is this: given a set of

input vertices with corresponding noisy outputs, predict the underlying function on the vertices of the graph that generated the outputs. The solution to this problem using GPs has been well studied for generic input spaces [96] and is easily applied to the graph case. There is, however, a less well studied problem for GP regression that cannot be satisfactorily generically addressed across all input spaces: what is the average case performance of the method as a function of the number of examples, or, what is the *learning curve*? The aim of this thesis is to address this problem on the subset of input spaces consisting of graph ensembles.

With the ever increasing amount of data we generate in our daily lives and, perhaps more importantly, the increase in our ability to store this data for later analysis, many techniques have been developed to make inference upon data defined on graphs. One such method was developed by Kondor and Lafferty [62] and then later improved upon by Smola and Kondor [109]. These authors introduced a series of kernels that could measure similarities between vertices of a graph using the normalised graph Laplacian (the discrete analogue of the Laplacian in continuous spaces [25]). In this thesis I will study the use of one particular kernel from this class known as the *random walk kernel* [109], applied in the context of Gaussian processes for regression.

There are many examples of real valued data with structure described by a graph, both man made and from nature. Some examples are collaboration networks, where vertices are authors of papers, edges connect authors who have collaborated with each other and the function values at each vertex indicates of an author's productivity and collaborativeness [81]; topic graphs, where vertices correspond to title words of documents, edges correspond to titles that contain both topics and the function on the vertices is defined to be the total number of mentions of the topic [117]; and finally image analysis where vertices in the graph correspond to pixels in the image, edges correspond to neighbouring pixels and function values are the pixel intensity at that vertex [63]. Being able to approximate the error of a method on these graphs would enable one to make statements about the expected number of examples of function values required to make predictions about the function defined on the graph to within a desired accuracy.

Perhaps a more useful property, however, is not to calculate the error for each instance of a graph and a function defined on it but the average performance of a GP averaged

over graphs and functions sampled from a particular distribution, giving the average generalisation error that defines the learning curve. It has been noted that many graphs that result from a range of problems from interactions within a cell to social networks to the world wide web exhibit common topological features [9] and that these in turn influence properties defined at the vertices of the graph [107, 119, 138]. Since these properties are common amongst all these graphs it is more useful then to address the problem of measuring the performance of a GP for regression on graphs by looking at its performance averaged over all graphs that share common topological properties. This idea will be the main focus of this thesis. I aim to create a method for accurately predicting the expected performance of GPs for learning on graphs which share common topological features.

Ever since the seminal work of Seung et al. [101], it has been recognised that statistical physics can make significant contributions to the understanding of methods and algorithms that learn from examples. Learning curves have been successfully analysed using statistical physics for a variety of parametric learning methods (where a finite number of parameters must be learned) by taking advantage of the interpretation of the average over training sets as quenched disorder [5, 19, 49, 87, 101, 130]. Rather more challenging is the non-parametric case, where the number of parameters is effectively infinite. An important example in this class is provided precisely by GPs. Here the set of parameters to be learned is in essence the entire underlying function that one is trying to estimate from the data. For the case of the average performance of GPs learning on random graph ensembles one can see that once more we are looking at a system under quenched disorder. This time we can relate a GP learning on a graph to a spin system on a lattice. In this system each site is a vertex in a graph, the function value is the 'spin' of the site and the quenched disorder comes from how the sites or vertices interact i.e. the edges within the graph within the ensemble of graphs we study.

The structure of the remainder of this thesis is as follows. In Chapter 2 I will discuss the background material underpinning the results to be derived in later chapters. Chapter 2 will also aim to give the reader an idea of the context of the results by detailing the results of other relevant research. After discussing all the relevant background material, in Chapter 3 I will then move on to discussing the graph kernels. The kernel is the

measure used by GPs to quantify the similarity between vertices on the graph. I will show that the random walk kernel [62, 109], which will be the focus of this thesis, has many unexpected properties including a non-trivial limiting form for large kernel length-scales. I will also show in this chapter that the typical way to normalise a kernel, which I will term global normalisation, leads to a probabilistically implausible model for the prior over data, I will propose a solution to this, which I term local normalisation.

After discussing in some detail the random walk kernel I will then present the main results of this thesis. In Chapter 4 I will derive the approximation of the learning curve of a GP under the assumption that the function to be predicted was generated from the GP prior, known as the 'matched case'. I will show by the application of the cavity method that it is possible to accurately predict the learning curves for ensembles of graphs that are constrained by their degree distribution for both globally and locally normalised kernels. We will see that predicting the learning curve of a GP with a locally normalised kernel presents a technically more difficult problem compared to the globally normalised kernel and that some further approximations than for the case of the globally normalised kernel are needed to get the final approximation. To understand these approximations and to clarify the interpretation of the predictions, in Chapter 5 I will derive the learning curves a second, equivalent way, using the replica method. We will see that by deriving the approximation of the learning curves using the replica method under the assumption of replica symmetry, the additional approximation required to calculate the locally normalised kernel amounts to ignoring some consistency constraints between neighbours on the graph.

The final set of results I will present will be given in Chapter 6. In this chapter I will consider the case of learning curves for GPs with model mismatch. Here one assumes that the function to be learnt was not generated by the GP prior being used to estimate the function. I will consider the case of functions being generated by another GP with a random walk kernel, but different hyperparameters and noise level. I will show by once more applying the cavity method that one can derive accurate predictions of the learning curves. We will see that in the presence of mismatch, additional features appear in the learning curve including regimes of overfitting. This can result in the learning curve having a peak lying above the initial error of the GP in the absence of any examples.

Finally in Chapter 7 I will summarise the results I have presented and discuss potential improvements and extensions. I will show that although I have made large inroads into the problem of predicting the learning curves of GPs on large graphs, there is scope for much further work.

CHAPTER 2

THEORETICAL BACKGROUND

In this chapter we will cover some of the main mathematical techniques used throughout this thesis. We will begin in section 2.1 by describing relevant material about GPs for regression. In section 2.1.1.1 to section 2.1.1.3 we will derive GPs for regression from a machine learning perspective: as a logical extension to linear regression. After the basic introduction given in section 2.1.1.3, in the rest of section 2.1 we will then discuss specific details about GP regression required to derive the results in subsequent chapters.

We will cover in section 2.1.2 specific properties about the kernels or covariance functions used for GPs for regression. Kernels are the mechanism in which a GP calculates similarity between examples. We will focus in particular on deriving the graph kernels introduced in Kondor and Lafferty [62], and further developed in Smola and Kondor [109], that calculate similarities between vertices of a graph.

After the discussion of kernels and their use for GPs for regression on graphs, in section 2.1.3, we will look at other methods proposed for the purpose of learning functions on graphs. Finally to conclude the background material in relation to GPs, section 2.1.4 will discuss methods that are used to characterise the performance of GPs for machine learning.

In later chapters we derive results using methods at the intersection between machine learning and statistical physics. In section 2.2 and section 2.3, we will therefore detail two of the main methods borrowed from statistical physics to derive later results. Section 2.2 will introduce the replica method. This will be approached by example, using the calculation by Malzahn and Opper [72] to explain the method. This introduction will serve a dual purpose, by also introducing an earlier approximation of the learning curves of GPs for regression that will be used as a baseline for the results derived in later chapters. Section 2.3 introduces the cavity method, also known as the *belief propagation* (BP) algorithm. It will be introduced using *undirected graphical models* or *Markov random fields*. This section will also briefly relate the method back to its roots in the statistical physics community.

Finally in section 2.4 we will consider graph ensembles. Graph ensembles will allow us to consider the performance of GP regression on classes of graphs with common topological features, such as a fixed but arbitrary degree distribution.

2.1 Gaussian Processes

GP regression in the machine learning literature has been derived as the limiting form of a number of, seemingly independent, techniques. These include extensions to RBF networks [80], spline smoothing [60, 128] or the basic Bayesian linear regression model. I will detail the latter derivation in what follows. Before we begin, however, it will first be useful to state the general regression problem formally:

Problem 2.1.1. Regression Problem: Given N inputs $\boldsymbol{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N)$ with $\boldsymbol{x}_i$ from the input space X and corresponding corrupted outputs $\boldsymbol{y} = (y_1, \ldots, y_N)^{\mathrm{T}}$ with $y_\mu \in \mathbb{R}$ generated by $y_\mu = f(\boldsymbol{x}_\mu) + \eta_\mu$ for $f : X \to \mathbb{R}$ some unknown function and η_μ some unknown *independently and identically distributed* (i.i.d.) random noise, find an estimate $\hat{f}$ of f.

The regression problem has been considered in many fields [see 20, 30, 36, 102, 128] for a variety of input spaces, X, both continuous and discrete. We will focus on just one method, GP regression. To derive GP regression I will begin by considering the *Bayesian linear regression* solution to problem 2.1.1 focussing in particular on the case

where the input space, X, is $\mathbb{R}^n$.

2.1.1 From linear regression to Gaussian process regression

Bayesian linear regression attempts to solve the regression problem by parametrising the function, f, using a finite set of independent *basis functions*. This in effect estimates the function f using functions that are parameterised by the basis functions, and is therefore known as a *parametric method*. We will show by the end of this section that by generalising to the case where we have an infinite number of basis functions we can derive GP regression [10, 20, 96]. We will start by following the Bayesian linear regression derivation given in Bishop [20], to extend this to GPs we will then follow the derivation in Barber [10].

2.1.1.1 Bayesian linear regression

Simple Bayesian linear regression begins by making a few stronger assumptions about the data presented to us than in problem 2.1.1. We constrain the regression problem in two ways: first we assume that the function we wish to calculate, f, belongs to, or can be accurately approximated by some family of functions that can be represented as linear combinations of linearly independent *basis functions* $\boldsymbol{\phi}(\boldsymbol{x}) = (\phi_1(\boldsymbol{x}), \ldots, \phi_M(\boldsymbol{x}))^{\mathrm{T}}$ with $\boldsymbol{\phi} : \mathbb{R}^n \to \mathbb{R}$, i.e. we assume f can be parameterised by $\boldsymbol{\phi}$ with $f(\boldsymbol{x}) = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{\phi}(\boldsymbol{x})$ and $\boldsymbol{w} \in \mathbb{R}^M$. Secondly, we assume that the corrupting noise, η_μ, is i.i.d. zero mean Gaussian noise with variance σ^2.

Subject to these two assumptions the probability or *likelihood* of the outputs $\boldsymbol{y}$ given the inputs $\boldsymbol{X}$ for a function parametrised by $\boldsymbol{w}$ will be

$$p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{w}) = \frac{1}{(2\pi\sigma^2)^{N/2}} \exp\left(-\frac{1}{2\sigma^2}\sum_{\mu=1}^{N}(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{\phi}(\boldsymbol{x}_\mu) - y_\mu)^2\right). \tag{2.1.1}$$

This is a product of independent Gaussians with means given by the function f evaluated at the inputs $\boldsymbol{x}_\mu$.

We would like to combine (2.1.1) with prior knowledge, for instance typical scale or smoothness properties of the function. Within a Bayesian framework this is achieved

using Bayes' theorem,

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}. \tag{2.1.2}$$

As we will see, by taking a Bayesian approach as opposed to a frequentist approach Bayes' theorem will enable us to not only produce estimates for $\hat{f}$ but also to give a measure of the uncertainty of this estimation. For this simple Bayesian linear regression derivation we will assume a-priori that the parameters or *weights* $\boldsymbol{w}$ are not too large. This can be encoded by a *prior* distribution $P(\boldsymbol{w})$. To make calculations easy (and to derive a GP in the limit of a large number of basis functions later) we will encode this belief by assuming weights $\boldsymbol{w}$ are Gaussian distributed about $\mathbf{0}$,

$$p(\boldsymbol{w}) = \left(\frac{\alpha}{2\pi}\right)^{M/2} \exp\left(-\frac{\alpha}{2}\boldsymbol{w}^{\mathrm{T}}\boldsymbol{w}\right). \tag{2.1.3}$$

Using Bayes' theorem (2.1.2), we combine prior knowledge and the likelihood of the data to define a *posterior* distribution over weights, $\boldsymbol{w}$, given the inputs, $\boldsymbol{X}$, and outputs, $\boldsymbol{y}$. Since both the prior and the likelihood are Gaussian distributions, the posterior will be a Gaussian distribution of the form,

$$\begin{aligned} p(\boldsymbol{w}|\boldsymbol{X},\boldsymbol{y}) &= \frac{p(\boldsymbol{y}|\boldsymbol{X},\boldsymbol{w})p(\boldsymbol{w})}{\int \mathrm{d}\boldsymbol{w}' p(\boldsymbol{y}|\boldsymbol{X},\boldsymbol{w}')p(\boldsymbol{w}')} \\ &= \frac{1}{(2\pi)^{M/2}|\boldsymbol{S}_N|^{1/2}} \exp\left(-\frac{1}{2}(\boldsymbol{w}-\boldsymbol{m}_N)^{\mathrm{T}}\boldsymbol{S}_N^{-1}(\boldsymbol{w}-\boldsymbol{m}_N)\right) \end{aligned} \tag{2.1.4}$$

where $\boldsymbol{S}_N = (\frac{1}{\sigma^2}\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi} + \alpha\boldsymbol{I})^{-1}$ for $\boldsymbol{I}$ the identity matrix and $\boldsymbol{m}_N = \frac{1}{\sigma^2}\boldsymbol{S}_N\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{y}$ with $\Phi_{ij} = \phi_j(\boldsymbol{x}_i)$. The posterior distribution can be interpreted a measure of the probability that a function with parameter $\boldsymbol{w}$ was responsible for generating $\boldsymbol{y}$ given our prior assumptions about the weights $\boldsymbol{w}$.

Equation (2.1.4) only tells us about the probability of a particular function being responsible for the data. Ultimately, however, we want to solve problem 2.1.1. To do this we must be able to make predictions about an unseen output y^* for input $\boldsymbol{x}^*$. In a Bayesian framework predictions are calculated using the *predictive distribution*, which is given by the likelihood of y^* given $\boldsymbol{w}$ averaged over the posterior distribution. Due to the simple forms of the prior and likelihood this is once more a Gaussian and is given by

$$\begin{aligned} p(y^*|\boldsymbol{X},\boldsymbol{y},\boldsymbol{x}^*) &= \int \mathrm{d}\boldsymbol{w}\, p(y^*|\boldsymbol{x}^*,\boldsymbol{w})p(\boldsymbol{w}|\boldsymbol{X},\boldsymbol{y}) \\ &= \frac{1}{(2\pi\sigma_*^2)^{1/2}} \exp\left(-\frac{1}{2\sigma_*^2}(y^* - \boldsymbol{m}_N{}^{\mathrm{T}}\boldsymbol{\phi}(\boldsymbol{x}^*))^2\right) \end{aligned} \tag{2.1.5}$$

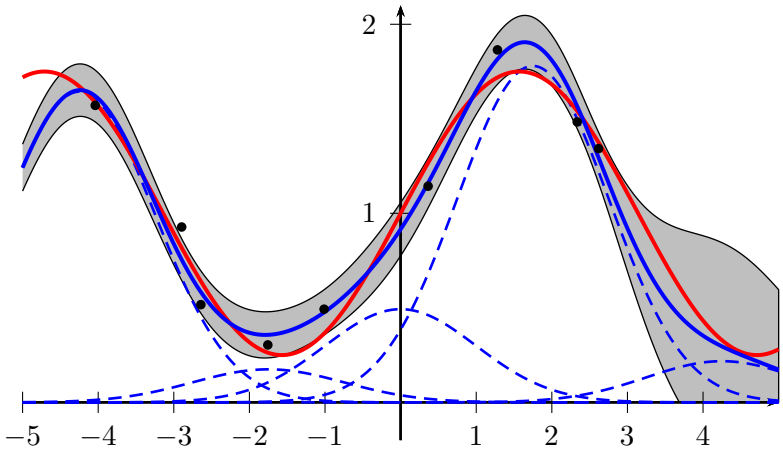


Figure 2.1: Plot to explain Bayesian linear regression. The mean of the predictive distribution (blue line) is just a linear weighting of the basis functions (dashed blue lines, taken to be Gaussian in this case) with weights given by $\boldsymbol{m}_N$. Dots represent $(\boldsymbol{x}_\mu, y_\mu)$ pairs and the red line represents the true underlying function. The shaded region shows one standard deviation of the predictive distribution from the mean.

where $\sigma_*^2 = \sigma^2 + \boldsymbol{\phi}(\boldsymbol{x}^*)^{\mathrm{T}} \boldsymbol{S}_N \boldsymbol{\phi}(\boldsymbol{x}^*)$.

If we assume a *squared loss function* i.e. mistakes are penalised by the square of the distance from the correct solution, then the 'best' solution of problem 2.1.1 is to take the mean of (2.1.5), i.e. we set $\hat{f}(\boldsymbol{x}) = \boldsymbol{m}_N{}^{\mathrm{T}} \boldsymbol{\phi}(\boldsymbol{x})$ [129]. This is known as the *Bayes predictor*. Since we took a Bayesian approach then as well as making predictions we can also give a measure of uncertainty in the prediction using the standard deviation of the predictive distribution, σ_*. This is one of the main advantages of this approach over other methods (see section 2.1.2 and section 2.1.3 for examples of other approaches).

Bayesian linear regression is illustrated in figure 2.1 for the case of Gaussian basis functions with unit variance shown with dashed blue lines and true underlying function $f(x) = 0.75 \sin(x) + 1$ indicated by a red line. The data plotted as black points is corrupted by Gaussian noise with variance $\sigma^2 = 0.01$. As the diagram shows the mean of the Bayesian prediction selects the best weights (in this case heights) of the basis functions in order to approximate the data it has been presented. The shaded region shows one standard deviation of the predictive distribution from the mean, it represents the error bar for every prediction point in the domain $[-5, 5]$. Figure 2.1 also emphasises one of the advantages of the Bayesian approach as opposed to frequentist approaches (like the

regularisation approach discussed in section 2.1.2). The full posterior distribution allows us to give an indication of the confidence in our prediction. This is especially useful in e.g. the region in the far right of the plot that has had no examples. The uncertainty in the prediction in this region is high giving us an indication that our estimate, $\hat{f}$, may be inaccurate.

2.1.1.2 Extending Bayesian linear regression: The function space view

Although useful, Bayesian linear regression has some problems; typically, we do not know what a 'good' parametrisation is for f. How then, do you pick the set of basis functions $\boldsymbol{\phi}(\boldsymbol{x})$? Even worse, what if X is not a subset of $\mathbb{R}^n$, what is the correct set of basis functions for this space?

One can begin to address these problems by instead considering GPs, the focus of this thesis. So far we have considered the prior (2.1.3) as a distribution over weights, $\boldsymbol{w}$, known as the *weight space view* [96]. However, taking a second, closer look at Bayesian linear regression we see that by parametrising the function and then placing the prior (2.1.3) over $\boldsymbol{w}$ we have actually placed a distribution over functions. One can also interpret the prior therefore as placing a Gaussian distribution over a function space spanned by the basis functions $\{\phi_1(\boldsymbol{x}), \ldots, \phi_M(\boldsymbol{x})\}$. This point of view is known as the *function space view* [96]. If we expand on this interpretation we see that (2.1.3) can also be thought of as a-priori placing a Gaussian distribution over the vector of function values, $\boldsymbol{f} = (f(x_1), \ldots, f(x_N))^{\mathrm{T}} = \boldsymbol{\Phi}\boldsymbol{w}$, with

$$\begin{aligned} \mathbb{E}[\boldsymbol{f}] &= \boldsymbol{\Phi}\mathbb{E}[\boldsymbol{w}] = \boldsymbol{0} \\ \mathrm{Cov}[\boldsymbol{f}] &= \mathbb{E}[\boldsymbol{f}\boldsymbol{f}^{\mathrm{T}}] = \boldsymbol{\Phi}\mathbb{E}[\boldsymbol{w}\boldsymbol{w}^{\mathrm{T}}]\boldsymbol{\Phi}^{\mathrm{T}} = \boldsymbol{C} \end{aligned} \tag{2.1.6}$$

where $C_{\mu\nu} = C(\boldsymbol{x}_\mu, \boldsymbol{x}_\nu)$ with *kernel* $C(\boldsymbol{x}_\mu, \boldsymbol{x}_\nu) = \frac{1}{\alpha}\boldsymbol{\phi}(\boldsymbol{x}_\mu)^{\mathrm{T}}\boldsymbol{\phi}(\boldsymbol{x}_\nu)$. This is a particular (restrictive) example of a GP (see definition 2.1.1) by extending linear regression to arbitrary positive definite operators we can address some of the problems with Bayesian linear regression.

2.1.1.3 Gaussian processes for regression

A GP is defined as follows;

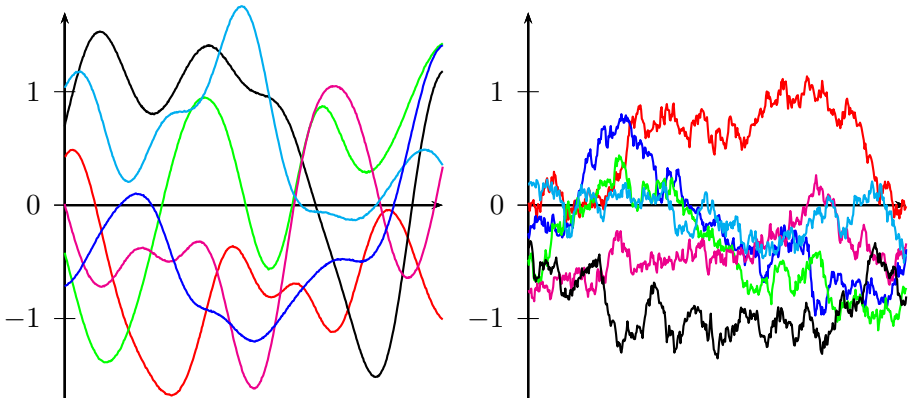


Figure 2.2: Figure showing 6 samples of functions from typical GP priors. (left) The squared exponential kernel (see table 2.1) with lengthscale $\sigma^2 = 0.1$. (right) The Ornstein-Uhlenbek kernel (see table 2.1) with lengthscale $\sigma = 0.5$.

Definition 2.1.1 (Gaussian Process)**.** A stochastic process $f(\boldsymbol{x})$ is called a *Gaussian process* (GP) with *covariance function* or *kernel* $C(\boldsymbol{x}, \boldsymbol{x}')$ and *mean function* $m(\boldsymbol{x})$, if for any finite number of points $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, the set of points $\{f(\boldsymbol{x}_1), \ldots, f(\boldsymbol{x}_N)\}$ is Gaussian distributed with covariance matrix $\boldsymbol{K}$ given by $K_{\nu\mu} = C(\boldsymbol{x}_\nu, \boldsymbol{x}_\mu)$ and mean vector $\boldsymbol{m}$ given by $m_\nu = \mu(\boldsymbol{x}_\nu)$.

It becomes clear from definition 2.1.1 that Bayesian linear regression is really just using a restricted class of GPs to represent f. Our assumptions in section 2.1.1.1 that gave (2.1.6) can be viewed as a long winded way of saying a-priori that the function is a sample from a GP with covariance and mean functions given by (2.1.6). This realisation makes the path from parametric regression to non-parametric regression clear; instead of parametrising f and defining basis functions we could assume a-priori that f was a sample from a GP with given kernel and mean functions.

By introducing GPs we will be able to place a distribution over a far broader space of functions. As an added benefit, because by framing the problem in terms of GPs we have abstracted away the parametrisation of f, we are able to place GPs over arbitrary input spaces, X, so long as the covariance function we use is a positive semi-definite operator. Examples of functions drawn from typical GP priors are given in figure 2.2 and figure 2.3.

With our new GP interpretation of the prior we can return once more to the regression

problem (problem 2.1.1). We can now solve this problem using the framework of GP regression whilst avoiding having to explicitly parameterise the function. We will assume once more that the data is corrupted by i.i.d. Gaussian noise with variance σ^2 thus keeping the likelihood in the form of (2.1.1). This time however, we will assume that the prior is given by a GP with covariance function $C(\boldsymbol{x}, \boldsymbol{x}')$ and mean function equal to zero. Under these assumptions, the marginal likelihood over outputs $\boldsymbol{y}$ will be given by

$$\begin{aligned} p(\boldsymbol{y}|\boldsymbol{X}) &= \int \mathrm{d}\boldsymbol{f}\, p(\boldsymbol{y}|\boldsymbol{f}, \boldsymbol{X}) p(\boldsymbol{f}|\boldsymbol{X}) \\ &= \frac{1}{(2\pi)^{N/2}|\boldsymbol{K}|^{1/2}} \exp\left(-\frac{1}{2}\boldsymbol{y}^{\mathrm{T}}\boldsymbol{K}^{-1}\boldsymbol{y}\right) \end{aligned} \tag{2.1.7}$$

This is a Gaussian with $\boldsymbol{f} = (f(\boldsymbol{x}_1), \ldots, f(\boldsymbol{x}_N))^{\mathrm{T}}$ and covariance matrix $K_{\mu\nu} = C(\boldsymbol{x}_\mu, \boldsymbol{x}_\nu) + \delta_{\mu\nu}\sigma^2$. The posterior will be a GP [96] with mean function

$$\bar{f}(\boldsymbol{x}) = \boldsymbol{k}(\boldsymbol{x})^{\mathrm{T}}\boldsymbol{K}^{-1}\boldsymbol{y} \tag{2.1.8}$$

and covariance function

$$\mathrm{Cov}(\boldsymbol{x}, \boldsymbol{x}') = C(\boldsymbol{x}, \boldsymbol{x}') - \boldsymbol{k}(\boldsymbol{x})^{\mathrm{T}}\boldsymbol{K}^{-1}\boldsymbol{k}(\boldsymbol{x}') \tag{2.1.9}$$

where we have defined $(k(\boldsymbol{x}))_\nu = C(\boldsymbol{x}_\nu, \boldsymbol{x})$.

In direct analogy with the Bayesian linear regression solution we require the predictive distribution to make estimates of function values at new input points. This is constructed by considering the joint distribution between outputs $\boldsymbol{y}$ and y^*. We see that the joint distribution will be of the form given in (2.1.7). To evaluate the predictive distribution we condition on $\boldsymbol{y}$ to give

$$p(y^*|\boldsymbol{X}, \boldsymbol{y}, \boldsymbol{x}^*) = \frac{1}{(2\pi\sigma_*^2)^{1/2}} \exp\left(\frac{1}{2\sigma_*^2}(y^* - m(\boldsymbol{x}^*))^2\right), \tag{2.1.10}$$

a Gaussian distribution with variance $\sigma_*^2 = C(\boldsymbol{x}^*, \boldsymbol{x}^*) + \sigma^2 - \boldsymbol{k}(\boldsymbol{x}^*)^{\mathrm{T}}\boldsymbol{K}^{-1}\boldsymbol{k}(\boldsymbol{x}^*)$ and mean $m(\boldsymbol{x}^*) = \boldsymbol{k}(\boldsymbol{x}^*)^{\mathrm{T}}\boldsymbol{K}^{-1}\boldsymbol{y}$. Assuming a squared loss function the best prediction is once more the Bayes predictor and is given this time by the mean, $m(\boldsymbol{x}^*)$, of (2.1.10). This is identical to (2.1.8) so can also be interpreted as the average over all functions weighted by the posterior distribution.

By introducing GPs we have solved to some extent the problems of Bayesian linear regression. Using a GP prior has allowed us to consider a larger space of functions

than in the linear regression case and eliminated, to a degree, the problem of how to parametrise f (although we now have gained the problem of picking the covariance function). Further still, by using a GP prior we have been able to generalise to non-Euclidean spaces. All of this however, has come at a cost: GP regression at its heart requires the inverse of $\boldsymbol{K}$, an $O(N^3)$ operation as opposed to the calculation of the inverse $S_N = (\frac{1}{\sigma^2}\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi} + \alpha\boldsymbol{I})^{-1}$, an $O(M^3)$ operation. As the number of examples, N, becomes large GPs will eventually become computationally infeasible and approximation methods will be required [see for example 96].

It is worth noting here that, as is standard in the literature, we will always assume, for simplicity, that the GP prior has a mean function equal to zero. Results that we will derive can easily be generalised to a non-zero mean. Introducing the mean function simply makes the resulting equations unwieldy and should only be contemplated when one has rather strong prior knowledge. For a discussion on GP regression with non-zero mean see for example Rasmussen and Williams [96].

2.1.2 Kernels

Now that we have abstracted away the parameterisation of the function using a kernel some of the basic insight about the prior discussed in section 2.1.1.1 has been lost. We can regain some of this insight about how a GP prior is affected by the choice of the kernel, however, by following Williams [132] and applying Mercer's theorem [135]. Mercer's theorem allows us to interpret the kernel as describing an unknown, possibly infinite dimensional, *Reproducing Kernel Hilbert Space* (RKHS) [8] (defined shortly below) comprised of basis functions given by the kernel eigenfunctions $\phi_\lambda(\boldsymbol{x})$. The inner products between eigenfunctions or feature vectors $\boldsymbol{\phi}(\cdot)$ of two inputs $\boldsymbol{x}$ and $\boldsymbol{x}'$ in this space are used to describe correlations between outputs in a similar manner to the basis functions in the Bayesian linear regression case. The GP regression solution of problem 2.1.1 therefore can be used for any input space X over which we can place a RKHS, a large number of input spaces indeed.

The introduction of GPs for regression as a way of eliminating the problem of parametrisation is known as the 'kernel trick'. Kernels allowed us to use an infinite number of basis functions while avoiding the problem of dealing with them directly, a very clever

trick. Implicitly, kernels calculate inner products of vectors of basis functions without direct computation.

Kernels are a large diverse area of research that cannot be covered in great detail in this thesis [see for example 104, and references therein]. We will discuss here the relevant results and derivations for what follows in subsequent chapters. We will begin with a discussion of the relationship between kernels and regularisation. In so doing we will be able to derive the *random walk kernel* in a similar manner to that seen in Smola and Kondor [109].

Section 2.1.1.1 introduced a Bayesian approach for solving the regression problem using prior knowledge. There are, of course, many other ways to solve this problem. One method is by *regularisation*. The aim of regularisation is to find the function f that minimises the functional

$$J[f] = \frac{\lambda}{2}\|Lf\|_{L^2}^2 + C(f, \boldsymbol{y}) \tag{2.1.11}$$

for some fixed *cost function* $C(f, \boldsymbol{y})$ and scaling parameter $\lambda \in \mathbb{R}^+$. The first term on the *right hand side* (RHS) of (2.1.11) is known as the *regularisation term* and represents prior assumptions about f, typically smoothness properties. In the literature, L is known as the *regularisation operator* and must satisfy the condition that it is an operator mapping from a Hilbert space $\mathcal{H}$ of functions f, to an inner product space with inner product $\langle Lf, Lg \rangle$ with $f, g \in \mathcal{H}$.

One can see the relationship between regularisation and GPs by studying (2.1.11) with a quadratic cost function $C(f, \boldsymbol{y}) = \frac{1}{2\sigma^2}\sum_i (f(x_i) - y_i)^2$. With this cost function we see that the minimum of (2.1.11) corresponds to finding the maximum of

$$\exp\left(-J[f]\right) = \exp\left(-\frac{\lambda}{2}\|Lf\|_{L^2}^2\right)\exp\left(-\frac{1}{2\sigma^2}\sum_i (f_i - y_i)^2\right). \tag{2.1.12}$$

This is just the *maximum a-posterior* (MAP) solution or mode of the GP with a kernel defined indirectly using L. Since, for a Gaussian, the mode and mean coincide, this is the optimum solution found when calculating the Bayes predictor (equation (2.1.10)) for some, as yet, unknown kernel (note that we do not have the added benefit of a distribution over posterior outputs). In order to study this relationship between regularisation and GPs for regression further we will require a definition.

Definition 2.1.2 (*Reproducing Kernel Hilbert Space* (RKHS))**.** Let $\mathcal{H}$ be a Hilbert space of real functions f mapping from X. Then $\mathcal{H}$ is a *Reproducing Kernel Hilbert Space* (RKHS) with inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ if there exists a function $k : X \times X \to \mathbb{R}$ so that

1. $\forall \boldsymbol{x}, k(\boldsymbol{x}, \cdot) \in \mathcal{H}$
2. $\langle f(\cdot), k(\cdot, \boldsymbol{x}) \rangle_{\mathcal{H}} = f(\boldsymbol{x})$

With this definition we will be able to say even more about the relationship between the MAP solution of GPs for regression and the regularisation method. By examining this relationship we will see that the suggestions made by Kondor and Lafferty [62], Smola and Kondor [109] for kernels on graphs are simply the natural extension of Euclidean space kernels.

2.1.2.1 From the squared exponential kernel to the random walk kernel

Kernels have been developed for a wide range of spaces [see for example 46] with a significant proportion focussing on Euclidean spaces, $\mathbb{R}^n$. Amongst the most frequently used kernels in continuous spaces is the *squared exponential kernel*

$$C(\boldsymbol{x}, \boldsymbol{x}') = \exp\left(-\frac{1}{2\sigma^2}\|\boldsymbol{x} - \boldsymbol{x}'\|^2\right). \tag{2.1.13}$$

The squared exponential kernel has been shown by Kimeldorf and Wahba [60], Smola and Schölkopf [110], Yuille and Grzywacz [137] to be one of a family of translationally invariant positive definite kernels that can be seen as regularising in some way using linear differential operators.

The relationship between kernels of GPs and regularisers with differential operators was first detailed in Kimeldorf and Wahba [60]. In Kimeldorf and Wahba [60] the authors were concerned with relating GP regression for inputs from the space $\mathbb{R}$ with the problem of regularising using the differential operator on $\mathbb{R}$

$$L = \sum_{i=0}^{m} a_i D^i. \tag{2.1.14}$$

for $a_i \in \mathbb{R}$ and $D^i = \left(\frac{d}{dx}\right)^i$ the i-th differential operator.

The solution to the regularisation problem can be shown to be an L-spline with connections, or *knots*, at the examples $y_1, \ldots, y_N$. An L-spline is a $(2m-2)$-th differential piecewise function which satisfies $L^*Lf = 0$ between knots where L^* is the adjoint of L given by,

$$L^* = \sum_{i=0}^{m} a_i(-D)^i \tag{2.1.15}$$

Kimeldorf and Wahba [60] wanted to find the equivalent kernel for a GP prior that gave a posterior MAP solution equal to the L-spline solution

The authors showed that for a GP prior with zero mean function and kernel given by

$$k(x, x') = (2\pi)^{-1} \int_{-\infty}^{\infty} \mathrm{d}\lambda \, \mathrm{e}^{\mathrm{i}(x-x')\lambda} |P(\lambda)|^{-2}, \quad P(\lambda) = \sum_{i=0}^{m} a_i(\mathrm{i}\lambda)^i \tag{2.1.16}$$

the MAP solution is equivalent to minimising (2.1.11) with squared loss function and regulariser given by (2.1.14) with knots at the example points (x_μ, y_μ). In other words, the MAP solution for GP regression with continuous kernels in the form of (2.1.16) gives $2m-2$ differentiable splines.

More recently this result has been independently proved by Poggio and Girosi [95], Yuille and Grzywacz [137] for the related case

$$\|Lf\|_{L^2}^2 = \sum_{i=0}^{m} b_i \|D^i f\|_{L^2}^2, \tag{2.1.17}$$

which can be viewed as specifying powers of the Laplacian operator. In these papers the authors approached the problem using Green's functions, which I will now summarise.

2.1.2.2 Relating regularisation to Gaussian processes

If we minimise (2.1.11) via variational methods we have

$$L^*Lf(x) = \frac{1}{\lambda} \sum_{i=1}^{m} (y_i - f(x_i))\delta(x - x_i) \tag{2.1.18}$$

where L^* represents the adjoint of L. We can solve (2.1.18) using a Green's function, $G(x, x')$, satisfying

$$L^*LG(x, x') = \delta(x - x'), \tag{2.1.19}$$

to get

$$f(x_i) = \sum_{j=1}^{m} G(x_i, x_j)\alpha_j, \qquad \alpha_j = \frac{1}{\lambda}(y_j - f(x_j)) \tag{2.1.20}$$

for $i = 1, \ldots, m$. The solution reads as

$$f(x) = \sum_{i=1}^{m} G(x, x_i)\alpha_i. \tag{2.1.21}$$

In its current form (2.1.20) looks odd because it contains $f(x_i)$, which is unknown. However, by rewriting we see that (2.1.20) defines the linear system

$$(\boldsymbol{G} + \lambda \boldsymbol{I})\boldsymbol{\alpha} = \boldsymbol{y} \tag{2.1.22}$$

for $G_{ij} = G(x_i, x_j)$ and $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_N)^{\mathrm{T}}$. Upon solving (2.1.22), we see that the dependence on $f(x_i)$ is eliminated and the final solution is given by

$$f(x) = \boldsymbol{g}^{\mathrm{T}}(\boldsymbol{G} + \lambda \boldsymbol{I})^{-1}\boldsymbol{y} \tag{2.1.23}$$

with $\boldsymbol{g} = (G(x, x_1), \ldots, G(x, x_N))^{\mathrm{T}}$.

The Green's function can easily be verified to be the kernel of an RKHS with squared norm $\|Lf\|_{L^2}^2 = \sum_{i=0}^{m} b_i \|D^i f\|_{L^2}^2$. Equation (2.1.23) therefore has the same form as the mean of the predictive distribution of a GP given by (2.1.10) with a covariance function, $\lambda G(x, x')$. From this we see the direct equivalence between GP regression and regularisation with splines i.e. the MAP solution of a GP with kernel $\lambda G(x, x')$ is the same as the minimum of (2.1.11) with regulariser given by (2.1.17).

All that remains to make this equivalence explicit is to calculate the form of G. This can be done using Fourier transforms. Taking the Fourier transform of both sides of (2.1.19) and reworking eventually results in

$$G(x, x') = \int \mathrm{d}s \, \frac{\mathrm{e}^{\mathrm{i}(x - x')s}}{\sum_m b_m s^{2m}}. \tag{2.1.24}$$

By this argument we see that by specifying a_i for the case of Kimeldorf and Wahba [60] or, as is more commonly found, specifying b_i in the case of Poggio and Girosi [95], one can derive kernel–regularisation pairs. One can show that both the ubiquitous squared exponential kernel and the Ornstein-Uhlenbeck kernel are two of a family of kernels that are paired with linear differential operator regularisers. For a comparison see table 2.1.

The relationship between kernels and differential operators allows us to construct covariance functions for GP priors that encode particular differentiability properties that the prior favours. Smola and Kondor [109] extended this idea to the case of functions defined on the vertices of graphs.

Kernel	Kernel Function	Regulariser Coefficients
Squared Exponential [see 137]	$\exp(-\frac{1}{2\sigma^2}\|x-x'\|^2)$	$b_i = \frac{\sigma^{2i}}{i!2^{2i}}$, $i = 0, \ldots, \infty$
Ornstein-Uhlenbeck	$\exp(-\frac{1}{\sigma}\|x-x'\|)$	$b_0 = 1, b_1 = \sigma^2,$ $b_i = 0$, $i = 2, \ldots, \infty$

Table 2.1: Various kernels and their corresponding differential operators for Euclidean spaces. Samples from these kernels are shown in figure 2.2

Graphs are objects that describe relationships between points or vertices using edges. They appear in many aspects of science to describe wide ranging relationships from cell biology to social interactions to traffic management. We define a graph as follows;

Definition 2.1.3 (Graph)**.** A graph $G(\mathcal{V}, \mathcal{E})$ is defined as a representation of relationships between objects or *vertices* from a vertex set $\mathcal{V}$. Relationships between two vertices are represented by a line or *edge* connecting them. The set of all edges in a graph G is called the edge set and is denoted by $\mathcal{E}$. A graph can also be represented by an *adjacency matrix* $\boldsymbol{A} \in \{0,1\}^{V \times V}$ where a_{ij} is one if $(i,j) \in \mathcal{E}$ and zero otherwise. We will assume an *undirected* graph so that the edge (i,j) is considered the same as the edge (j,i).

Smola and Kondor [109] were concerned with creating kernels which assumed that function values on the vertices of a graph were correlated through the edges of the graph. In terms of a regression problem the authors' kernels could be used to solve problem 2.1.1 with an input space X equal to the vertex set of a graph and functions f defined to be functions that map from vertices of a graph to the real line, $f : \mathcal{V} \to \mathbb{R}$. Smola and Kondor [109] chose to encode relationships between vertices using the *normalised graph Laplacian* [25] defined as

Definition 2.1.4 (Normalised Graph Laplacian [25])**.** The *normalised graph Laplacian* $\boldsymbol{L}$ for a graph $G(\mathcal{V}, \mathcal{E})$ with vertex set $\mathcal{V}$ and edge set $\mathcal{E}$ is defined as[1]

$$\boldsymbol{L}_{uv} = \begin{cases} 1 & \text{if } u = v \\ -\frac{1}{\sqrt{d_u d_v}} & \text{if } (u, v) \in \mathcal{E} \\ 0 & \text{otherwise} \end{cases} \tag{2.1.25}$$

for $u, v \in \mathcal{V}$ and d_u defined to be the degree of u (the number of edges connected to u).

The relationship between the normalised graph Laplacian and the Laplacian in continuous spaces can be realised by discretising the Laplacian of a function f in $\mathbb{R}^2$ by a regular two dimensional grid. In this situation the discretisation will give

$$\begin{aligned} Lf(x, y) = \Delta f(x, y) &= \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \\ &\approx \frac{f(x + \Delta t, y) + f(x - \Delta t, y) + f(x, y - \Delta t) + f(x, y + \Delta t) - 4f(x, y)}{(\Delta t)^2} \\ &= -\frac{4}{(\Delta t)^2} (\boldsymbol{L}\boldsymbol{f})_{(x,y)} \end{aligned} \tag{2.1.26}$$

where we have defined the $\boldsymbol{L}$ to be the normalised Laplacian on the graph given by the discretisation grid and $\boldsymbol{f}$ to be a function mapping from vertices on the grid to $\mathbb{R}$ with values equal to the original function at the grid points.

Using definition 2.1.4 as a discrete approximation to the Laplacian Smola and Kondor [109] constructed kernels in a similar manner to Poggio and Girosi [95], Yuille and Grzywacz [137]. These required the discrete equivalent of Green's functions [28]. Smola and Kondor [109] used discrete Green's functions to prove the following result

Theorem 2.1.1 (Discrete Green's functions [109])**.** Let $\boldsymbol{P} \in \mathbb{R}^{V \times V}$ be a positive semi-definite regularisation matrix and let $\mathcal{H}$ be the image of $\mathbb{R}^V$ under $\boldsymbol{P}$. Then $\mathcal{H}$ with dot product $\langle \boldsymbol{f}, \boldsymbol{P}\boldsymbol{f} \rangle$ is a RKHS and its kernel is $C(i, j) = (\boldsymbol{P}^{-1})_{ij}$, where $\boldsymbol{P}^{-1}$ is the pseudo-inverse if $\boldsymbol{P}$ is not invertible.

[1] This is a slightly different definition from that in [25] in that we do not set $L_{vv} = 0$ for $d_v = 0$. We do this to simplify later equations by eliminating the need for a special case distinction. This is justified since the change in definition only impacts disconnected single vertices and, in the end, we will argue for the locally normalised kernel (see Chapter 3) in which case the two definitions of the normalised Laplacian result in the same random walk kernel.

Kernel	Kernel Function	$r(\lambda)$
Regularised Laplacian	$(\boldsymbol{I} + \sigma^2 \boldsymbol{L})^{-1}$	$1 + \sigma^2 \lambda$
Diffusion	$\exp(-\frac{\sigma^2}{2}\boldsymbol{L})$	$\exp(\frac{\sigma^2}{2}\lambda)$
p-step Random Walk	$(\boldsymbol{I} - a^{-1}\boldsymbol{L})^p$	$(\boldsymbol{I} - a^{-1}\lambda)^{-p}$
Inverse Cosine	$\cos(\boldsymbol{L}\pi/4)$	$\sec(\lambda\frac{\pi}{4})$

Table 2.2: Various kernels for functions on graphs.

This simply states that the kernel is the discrete Green's function of $\boldsymbol{P}$. From this theorem we see that in a discrete space if

$$\boldsymbol{P} = r(\boldsymbol{L}) = \sum_i r(\lambda_i) \boldsymbol{v}_i{}^{\mathrm{T}} \boldsymbol{v}_i \tag{2.1.27}$$

for $\{(\lambda_i, \boldsymbol{v}_i)\}$ the eigenvalues and eigenvectors of $\boldsymbol{L}$ then

$$\boldsymbol{P}^{-1} = \boldsymbol{C} = r^{-1}(\boldsymbol{L}) = \sum_i r^{-1}(\lambda_i) \boldsymbol{v}_i{}^{\mathrm{T}} \boldsymbol{v}_i \tag{2.1.28}$$

Smola and Kondor [109] used (2.1.28) to introduce a range of kernels based on the choice of regulariser of the normalised graph Laplacian. The relationship between $r(\boldsymbol{L})$ and the continuous result given in (2.1.24) can be seen by writing r in terms of its power series. The most useful kernels are summarised in table 2.2. By comparing table 2.1 with table 2.2 it is easy to see the relationship between the graph kernels and their continuous counterparts.

This thesis will focus on the *random walk kernel* and its limiting form, the *diffusion kernel*, which can both be viewed as discrete versions of the popular squared exponential kernel. The diffusion kernel is the direct analogue of the squared exponential kernel, however, due to the numerical cost of calculating the diffusion process on graphs, the random walk kernel has been suggested as a suitable replacement [109]. This can be realised by keeping p/a constant with p and a large; in this situation the random walk kernel will approach the diffusion process. Samples from a GP prior with a random walk kernel are shown in figure 2.3. As p becomes larger, or alternatively a becomes smaller, the kernel lengthscale increases which results in the prior favouring smoother functions.

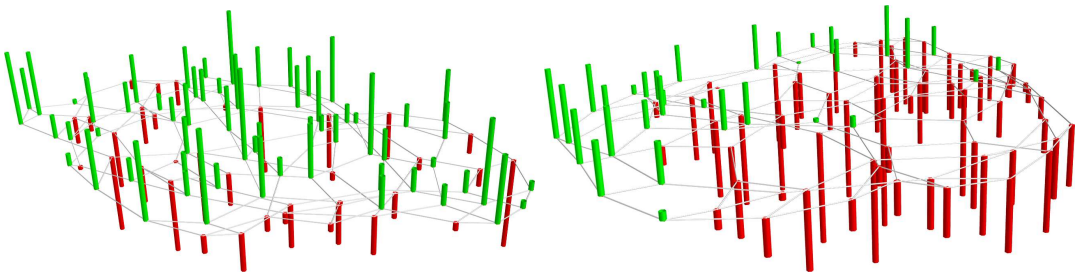


Figure 2.3: Samples from a GP prior with the random walk kernel on a graph with 100 vertices. (left) $p = 1$ and $a = 2$. (right) $p = 15$ and $a = 2$

2.1.3 Other methods for learning on graphs

Besides the set of kernels introduced by Kondor and Lafferty [62], Smola and Kondor [109] there have been many other attempts to perform learning on graphs, the majority of which use the (normalised) Laplacian in some way. The Laplacian is an obvious choice for such a task. As pointed out in Herbster et al. [52], if one wants to introduce a measure of the smoothness of a function, f, whilst preserving graph structure then an intuitive choice would be

$$S(f) = \frac{1}{2}\sum_{i,j} a_{ij}(f_i - f_j)^2. \tag{2.1.29}$$

One can easily see that for small fluctuations amongst neighbours (i.e. a slowly varying smooth function) we would have a small value for $S(f)$. Some basic algebra also shows that equation (2.1.29) corresponds to defining an inner product over the space of functions given by

$$\langle f, g\rangle_{\tilde{\boldsymbol{L}}} := f^{\mathrm{T}}\tilde{\boldsymbol{L}}g. \tag{2.1.30}$$

where $\tilde{\boldsymbol{L}} := \boldsymbol{D} - \boldsymbol{A}$ is defined to be the un-normalised Laplacian and where (2.1.29) corresponds to the induced semi-norm $\|f\|^2_{\tilde{\boldsymbol{L}}} := \langle f, f\rangle_{\tilde{\boldsymbol{L}}}$ (it is only a semi-norm because $\|f\|_{\tilde{\boldsymbol{L}}} = 0$ for all constant f) to measure smoothness.

The connection between function smoothness and the Laplacian can be exploited in a variety of ways. In Herbster and Pontil [51], Herbster et al. [52] it is used as a measure of 'distance' between functions for a perceptron learning algorithm. In later work [50] this is further generalised to use the semi-norm given by

$$\|f\|_p = \left(\sum_{i,j} a_{ij}|f_i - f_j|^p\right)^{1/p}. \tag{2.1.31}$$

where $p = 2$ corresponds to the un-normalised Laplacian.

A more direct application of the graph Laplacian was used by Belkin et al. [12]. Here the Laplacian is used as a regulariser in a similar manner to that discussed in section 2.1.2. In this case the regulariser is simply set equal to the un-normalised Laplacian. The authors also suggest, in a manner similar to Smola and Kondor [109], that one may consider powers of the Laplacian to derive a family of regularisers based on the un-normalised Laplacian.

In further work Belkin and Niyogi [11] noted that the eigenvectors of the un-normalised Laplacian are increasingly 'rough' as the corresponding eigenvalue increases. The authors suggested using functions from the space constructed from only p eigenvectors corresponding to the first p-th smallest eigenvalues of the Laplacian.

Zhu et al. [139] used the un-normalised graph Laplacian in a GP. In this paper the authors were interested in assigning labels to unlabelled data only and assumed that labelled data was correct (i.e. a zero noise GP) and pinned. The authors considered a *diffusion kernel* over the unlabelled vertices of the graph given by

$$\int_0^\infty \mathrm{d}t \, \exp(-t\tilde{\boldsymbol{L}}_{uu}) = (\boldsymbol{D}_{uu} - \boldsymbol{A}_{uu})^{-1} \tag{2.1.32}$$

with $\tilde{\boldsymbol{L}}_{uu}$ defined to be the submatrix of the Laplacian for unlabelled vertices in the graph.

Finally it is worth mentioning that a related problem to graph regression and classification is considered in Chapelle, Weston, and Schölkopf [24], Szummer and Jaakkola [120]. Here the authors use graphs as a means to perform semi-supervised learning in Euclidean spaces. The authors consider the situation where one is presented with only partially labelled data. A weighted graph is then constructed by assigning edges between data points with weights according to some measure (a squared exponential kernel in the case of Chapelle et al. [24], Szummer and Jaakkola [120]) with a cutoff for small values. Using this, regression or classification on the graph can be performed to make predictions for the unlabelled data. In the case of these papers the authors use random walks to assign correlations between vertices but it is trivial to apply some of the other kernels to achieve the same goal.

As is apparent from above, the (normalised) graph Laplacian provides a wealth of ker-

nels and other methods besides to perform regression or classification upon a graph. However, there are also many methods that can perform this task without the Laplacian as a measure of similarity. The most obvious of these is the k-nearest neighbour algorithm [102], where vertex labels are calculated using a weighted average of the k-nearest neighbours. More recently Herbster et al. [53] showed that, for the case of classification, Laplacian based techniques performed badly on large diameter graphs (the diameter is defined as the largest separation between any two vertices, where separation is defined as the length of the shortest path). Herbster et al. [53] proceeded to show that, by constructing a 'spine' or path from the graph using depth first exploration, they were in some way able to preserve structure, and use a 1-nearest neighbour method to achieve far better performance (see the next section for a discussion on how performance is measured).

2.1.4 Characterising the performance of Gaussian processes

For any prediction method to be of use we need to understand how well it generalises i.e. the accuracy of the method's predictions. One way of characterising the generalisation performance of a method is the *generalisation error*, given by the average error of the method over the entire space of inputs and outputs.

If we define $p(\boldsymbol{x}, y)$ to be the probability of the data pair $(\boldsymbol{x}, y)$ being generated by the teacher, then the generalisation error is given by,

$$\hat{\epsilon}_g = \int \mathrm{d}\boldsymbol{x}\mathrm{d}y\, p(\boldsymbol{x}, y)\mathcal{L}(f(\boldsymbol{x}), y), \tag{2.1.33}$$

for a loss function $\mathcal{L}$[2].

The generalisation error cannot, in general, be calculated during learning since typically one does not know $p(\boldsymbol{x}, y)$. One therefore has to estimate $\hat{\epsilon}_g$ in some way. The simplest approximation is given by the *training error*. The training error is the error the method makes at predicting the data D it has been presented. It is defined as

$$\hat{\epsilon}_t = \frac{1}{N}\sum_{\mu=1}^{N} \mathcal{L}(f(\boldsymbol{x}_\mu), y_\mu). \tag{2.1.34}$$

[2]Note that we have used $p(\boldsymbol{x}, y)$ here to distinguish the probability of a data pair being generated from that of the prior over data sets $P(\boldsymbol{x}, y)$.

In practice the training error is not very accurate and typically underestimates the generalisation error because of overfitting. Despite this, however, it can be used along with other information to provide good approximations of the generalisation error. One attempt to bound the generalisation error of Bayesian methods in this way is the *probably approximately correct – Bayesian* (PAC-Bayes) approach.

2.1.4.1 PAC-Bayes bounds

PAC-Bayes bounds are an attempt to extend frequentist approaches of measuring the performance of a method to Bayesian techniques. To understand the PAC-Bayes approach we must first briefly discuss the *probably approximately correct* (PAC) approach for characterising the performance of learning methods.

A PAC bound in its most basic form bounds the generalisation error of a method by using the training error, inputs $\boldsymbol{X}$, and *binary* outputs $\boldsymbol{y}$ (where outputs y are either zero or one). It asks, given a space of possible problems or functions on which the method must make predictions, $\mathcal{F}$, a data generating function, $p(\boldsymbol{x}, y)$, representing the unknown function in $\mathcal{F}$ to be learnt, some inputs, $\boldsymbol{X}$, and outputs, $\boldsymbol{y}$, generated i.i.d. from $p(\boldsymbol{x}, y)$, what is the smallest value, ϵ, such that $\hat{\epsilon}_g - \hat{\epsilon}_t \leq \epsilon$ at least $1 - \delta$ of the time with respect to repeated sampling of $\boldsymbol{y}$ and $\boldsymbol{X}$? More formally it is defined as follows

Definition 2.1.5 (PAC bound)**.** Let the set of all possible functions be given by $\mathcal{F}$, let f be the predicted function of a learning method from inputs $\boldsymbol{X}$ and outputs $\boldsymbol{y}$ drawn i.i.d. from an unknown distribution $p(\boldsymbol{x}, y)$. Let the confidence be given by δ. Find the smallest $\epsilon(\delta, \boldsymbol{X}, \boldsymbol{y}, f)$ such that

$$P\left(\hat{\epsilon}_g \leq \hat{\epsilon}_t + \epsilon(\delta, \boldsymbol{X}, \boldsymbol{y}, f)\right) \geq 1 - \delta \tag{2.1.35}$$

with

$$\hat{\epsilon}_g = \int \mathrm{d}\boldsymbol{x}\mathrm{d}y\, p(\boldsymbol{x}, y)\mathcal{L}(f(\boldsymbol{x}), y) \tag{2.1.36}$$

and

$$\hat{\epsilon}_t = \frac{1}{N}\sum_{i=1}^{N} \mathcal{L}(f(\boldsymbol{x}_i), y_i) \tag{2.1.37}$$

where P denotes probability over datasets drawn i.i.d. from $p(\boldsymbol{x}, y)$ and $\mathcal{L}$ represents a loss function.

Initially the study of PAC bounds was at odds with Bayesian approaches to machine learning; PAC bounds only assume that data is presented i.i.d. and calculates how well a method will perform at predicting all possible functions within a function space. This is typically a bound dominated by the complexity of the function space (given by its *Vapnik–Chervonenkis* (VC) dimension [123]) which grows as functions become more complicated. A Bayesian, however, incorporates as much as she knows and encodes this in a prior. She tunes the method so it is best suited to learning the problem. In this context the complexity of the space will play a smaller role in bounding the generalisation error. Thus, in order to achieve sensible bounds for the generalisation error of a Bayesian method using PAC techniques, a new method was needed. This method had to not rely solely on the VC dimension of the space and somehow take account of the prior placed over the space.

The first attempts to successfully apply PAC bounds to Bayesian methods focussed on binary classification tasks with loss functions, $\mathcal{L}(f(x), y) = (1 - \delta_{f(x),y})$. Unlike the earlier Bayes predictor given by the mean of (2.1.10) the authors considered the Gibbs predictor. The Gibbs predictor makes predictions for each new individual point by first sampling a function f from the posterior, and then making the prediction using this function. For each new prediction another sample is drawn from the posterior distribution. With the Gibbs predictor the generalisation error and training error for a posterior Q are given by

$$\hat{\epsilon}_g = \int \mathrm{d}\boldsymbol{x}\mathrm{d}y\, p(\boldsymbol{x}, y) \int \mathrm{d}f\, Q(f)[1 - \delta_{f(x),y}] \tag{2.1.38}$$

$$\hat{\epsilon}_t = \frac{1}{N} \sum_{i=1}^{N} \int \mathrm{d}f\, Q(f)[1 - \delta_{f(x_i),y_i}], \tag{2.1.39}$$

respectively. To apply PAC bounds to Bayesian methods authors initially considered the idea of structural risk minimisation [69, 105, 124, 125]. In this situation one assumes the learner is presented with a hierarchy of classes

$$H_1 \subseteq H_2 \subseteq \cdots \subseteq H_d \subseteq \cdots$$

and is told that f belongs to one of the classes H_d. The learner begins by looking for f in H_1 and then progresses through to the larger spaces until it finds a suitable function. One can account for the prior used in Bayesian learning by placing more probable functions in the smaller initial sets and the less probable functions in subsequent larger sets.

More recent PAC based bounds for Bayesian binary classifiers are based on the work of McAllester [74] and are derived from the earlier *structural risk minimisation* (SRM) PAC bounds considered in Shawe-Taylor et al. [105]. These PAC-Bayes bounds included a *Kullback-Leibler* (KL) divergence [67] between prior and posterior distribution given by

$$\mathrm{KL}[Q||P] = \int \mathrm{d}f\, Q(f) \left[\log \frac{Q(f)}{P(f)}\right]. \tag{2.1.40}$$

These bounds became tighter for priors that were 'closer' to the subsequently calculated posterior distribution and which therefore more accurately represented the true underlying data a-priori. The first PAC-Bayes bound derived by McAllester [74] was later improved by Langford [68], Seeger [100]. Our subsequent discussion about PAC-Bayes bounds for regression makes use of the result in Langford [68] so we will detail this here:

Theorem 2.1.2 (PAC-Bayes for classification [68])**.** For a function space of binary classifiers $\mathcal{F}$, any distribution p, prior P, posterior Q and confidence δ

$$P\left(\mathrm{KL}_{\mathrm{B}}(\hat{\epsilon}_t||\hat{\epsilon}_g) \leq \frac{1}{N}\left(\mathrm{KL}[Q\|P] + \log\frac{N+1}{\delta}\right)\right) < 1-\delta \tag{2.1.41}$$

with

$$\mathrm{KL}_{\mathrm{B}}(a||b) = \begin{cases} a\log\dfrac{b}{a} + (1-a)\log\dfrac{1-a}{1-b} & \text{for } b > a \\ 0 & \text{otherwise} \end{cases} \tag{2.1.42}$$

the KL divergence of two Bernoulli random variables.

Although this bounds the generalisation error of the Gibbs predictor one can also show that the generalisation error using the optimal Bayes predictior is bounded by twice the Gibbs error [100].

More recently there have been attempts at bounding the generalisation error of GP regression in a PAC-Bayes framework [103]. In this case the problem is recast into that of a classification task by creating classifiers from functions. These are created using an inverted zero-one loss function where the loss is one for a correct value and zero for an incorrect value. For a *zero noise* teacher this can be viewed as placing an ϵ_0 tube around the true function (see figure 2.4). Parts of f that are within the tube are labelled one and parts outside are labelled zero. With this formalism one can now consider generalisation

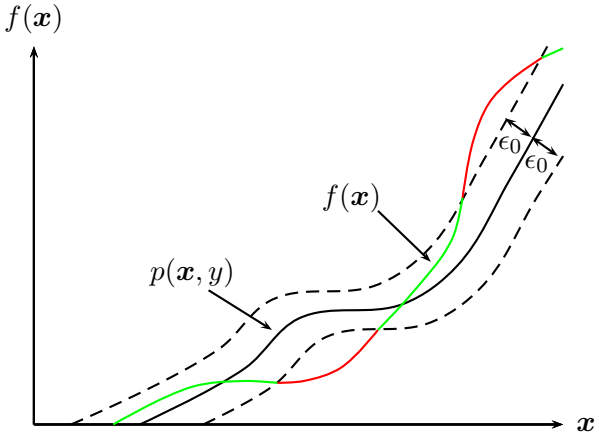


Figure 2.4: Diagram of the classifier $c_f(\boldsymbol{x}, y)$ for p without noise on outputs. For points of f inside the ϵ_0 tube the classifier returns a 0 (shown in green) for points lying outside the tube the classifier returns a 1 (shown in red).

and training errors given by

$$\hat{\epsilon}_g = \int \mathrm{d}\boldsymbol{x}\mathrm{d}y\, p(\boldsymbol{x}, y)\, P_{f\sim Q}(f(\boldsymbol{x}) \in (-\infty, y - \epsilon_0) \cup (y + \epsilon_0, \infty)) \tag{2.1.43}$$

$$\hat{\epsilon}_t = \frac{1}{N}\sum_{i=1}^{N} P_{f\sim Q}(f(\boldsymbol{x}_i) \in (-\infty, y_i - \epsilon_0) \cup (y_i + \epsilon_0, \infty)) \tag{2.1.44}$$

These measure the expected proportion of predictions, f, from the posterior, Q, lying outside of the ϵ_0 tube. Similar to the classification case, Shawe-Taylor [103] shows that one can derive a PAC-Bayes bound for $\hat{\epsilon}_g$ given by (2.1.43) using the training error given by (2.1.44). This bound is of the same form as (2.1.41). Thus by picking a tolerance for the predictions of the GP using ϵ_0 we can find a PAC-Bayes bound on the proportion of our prediction that lies outside of the tube.

Using (2.1.41) with training error (2.1.44) and setting P to be the GP prior and Q to be the GP posterior with mean (2.1.8) and covariance (2.1.9) we can bound the generalisation error (2.1.43) of GP regression for any given confidence δ. In other words we are able to state, for a given confidence δ, the maximum 'distance' of the generalisation error from the training error. Thus using this result we can determine with confidence δ, using the training error, $\hat{\epsilon}_t$, the point at which our GP model has seen enough examples, N, for the generalisation error, $\hat{\epsilon}_g$, to be below a given threshold without explicitly calculating $\hat{\epsilon}_g$.

2.1.4.2 Measuring average performance – learning curves

Although PAC-Bayes methods provide confidence based bounds on the generalisation error for a given set of input-output pairs, D, they can be rather loose. This looseness is necessary as the PAC-Bayes approach must consider the 'worst case' scenario with regard to the data presented to it. PAC-Bayes bounds in essence bound inference methods by finding the worst data set that the teacher can present to the student (with at least a probability δ of occurring) for learning a function f. Instead of a worst case approach, another method for giving an indication of a method's performance is to study the average case performance.

Average case performance is calculated by averaging the generalisation error over all data sets D of size N. This approach gives an idea of the typical performance of the method and, for large systems and data sets, will give a good indication of the actual performance of a particular instance of D [6]. As a function of N, the averaged generalisation error is known as the *learning curve*. We denote the average generalisation error by

$$\epsilon_g(N) = \left\langle \left\langle \int \mathrm{d}\boldsymbol{x}\mathrm{d}y\, p(\boldsymbol{x}, y)\mathcal{L}(\hat{f}(\boldsymbol{x}|\boldsymbol{X}, \boldsymbol{y}), y) \right\rangle_{\boldsymbol{y}|\boldsymbol{X}} \right\rangle_{\boldsymbol{X}} \tag{2.1.45}$$

for a method that predicts a function $\hat{f}$ given the inputs $\boldsymbol{X}$ and outputs $\boldsymbol{y}$.

Typically brute force numerical calculation of (2.1.45) is hard because sampling from the data distribution for large N is expensive, so approximations have to be made. One class of methods for approximating (2.1.45) uses mean field approaches from statistical mechanics. For parametric methods, approximations of this type have been found to be extremely accurate [5, 44, 49, 87, 101, 130]. Less, however, has been achieved for non-parametric methods such as GPs, with the majority of approximations, statistical mechanics based or not, either only qualitatively accurate for a broad range of cases [71, 72, 88, 111–113, 115, 116, 133], or quantitatively accurate but only for specific settings [72, 96].

In this thesis we will focus on trying to approximate the learning curve for GP regression defined on large random graphs with the random walk kernel (see table 2.2). As a baseline to compare new predictions against we will use the generalisation error approximations presented in Sollich [111] adjusted to the random graph case. These have been shown [72] to be a more restrictive case of the generalisation error approximation pre-

sented in Malzahn and Opper [72]. Before discussing the derivation in [72] we must first study in more detail the specific form of the generalisation error for GPs for regression.

In keeping with Sollich [111, 113], Sollich and Williams [116], I will derive approximations for a squared loss function $\mathcal{L}(\hat{f}(\boldsymbol{x}), y) = (\hat{f}(\boldsymbol{x}) - y)^2$ and a data set generated from a function g, sampled from a GP with covariance function $C_g(\boldsymbol{x}, \boldsymbol{x}')$ and zero mean, corrupted by some i.i.d. Gaussian noise with variance σ_g^2. With these assumptions the average generalisation error can be simplified and reads as

$$\epsilon_g(N) = \int \mathrm{d}\boldsymbol{x}\, p(\boldsymbol{x}) \Big(C_g(\boldsymbol{x}, \boldsymbol{x}) + \sigma_g^2 \\ - 2\boldsymbol{k}_f(\boldsymbol{x})^{\mathrm{T}} \boldsymbol{K}_f^{-1} \boldsymbol{k}_g(\boldsymbol{x}) + \boldsymbol{k}_f(\boldsymbol{x})^{\mathrm{T}} \boldsymbol{K}_f^{-1} \boldsymbol{K}_g \boldsymbol{K}_f^{-1} \boldsymbol{k}_f(\boldsymbol{x}) \Big) \quad (2.1.46)$$

where the subscripts f and g represent student and teacher parameters respectively, $(\boldsymbol{k}_f(\boldsymbol{x}))_\nu = C_f(\boldsymbol{x}_\nu, \boldsymbol{x})$ and $(\boldsymbol{K}_f)_{\mu\nu} = C_f(\boldsymbol{x}_\mu, \boldsymbol{x}_\nu)$ for $\mu, \nu = 1, \ldots, N$. For the case where the noise and prior of the student and teacher match, as was considered in Sollich [111, 112, 113], Sollich and Williams [116], Williams and Vivarelli [133], the generalisation error can be further simplified to give

$$\epsilon_g(N) = \int \mathrm{d}\boldsymbol{x}\, p(\boldsymbol{x}) \Big(C_f(\boldsymbol{x}, \boldsymbol{x}) + \sigma_f^2 - \boldsymbol{k}_f(\boldsymbol{x})^{\mathrm{T}} \boldsymbol{K}_f^{-1} \boldsymbol{k}_f(\boldsymbol{x}) \Big), \quad (2.1.47)$$

which is just the predictive variance of the GP (2.1.10) averaged over the distribution of inputs, $p(\boldsymbol{x})$.

To be able to extend earlier learning curve approximations to the random graph case an additional average must be included in the definition of the averaged generalisation error. We will study the generalisation error averaged over teachers g, inputs $\boldsymbol{X}$, outputs $\boldsymbol{y}$ (given inputs $\boldsymbol{X}$ and the teacher g) and finally over ensembles of graphs $\mathcal{G}$ (see section 2.4 for a discussion). As hinted at in the introduction and expanded upon in section 2.4 this will allow us to make statements about the performance of GPs for classes of random graphs with common features.

The generalisation error that we will try to predict in subsequent chapters will be of the form

$$\epsilon_g(N) = \Big\langle \Big\langle \Big\langle \Big\langle \int \mathrm{d}\boldsymbol{x}\, (\langle f(\boldsymbol{x}) \rangle_{f|\boldsymbol{y},\boldsymbol{X}} - g(\boldsymbol{x}))^2 \Big\rangle_{\boldsymbol{y}|g,\boldsymbol{X}} \Big\rangle_g \Big\rangle_{\boldsymbol{X}} \Big\rangle_{\mathcal{G}} . \quad (2.1.48)$$

for the case where student and teacher do not match, and

$$\epsilon_g(N) = \left\langle \left\langle \left\langle \left\langle \int \mathrm{d}\boldsymbol{x} \left(\langle f(\boldsymbol{x}) \rangle_{f|\boldsymbol{y},\boldsymbol{X}} - f(\boldsymbol{x}) \right)^2 \right\rangle_{\boldsymbol{y}|f,\boldsymbol{X}} \right\rangle_{f} \right\rangle_{\boldsymbol{X}} \right\rangle_{\mathcal{G}}, \tag{2.1.49}$$

for the case where the student's prior and teacher's GP match. This is also known as Bayes' error and is the lowest average error a GP can achieve.

In order to derive the approximation of the generalisation error given by Sollich [111] applied to random graphs using the methods presented in Malzahn and Opper [72] we will first need an introduction to statistical mechanics. We will begin the derivation of the baseline comparison, therefore, by describing the replica method.

2.2 Replica Method

The replica method was first proposed by Edwards and Anderson [37] in the field of theoretical physics to compute the free energy of the *Sherrington-Kirkpatrick* (SK) model. The SK model was a 'simple' model of a spin glass devised as a solvable version of the *Edwards-Anderson* (EA) model [37] of spins interacting on a lattice. It approximated the EA model by assuming infinite range interactions randomly distributed according to a Gaussian distribution with variance proportional to the inverse of the system size V. It was chosen because of its similarity to a non-disordered ferromagnet of which a solution was known.

Since the introduction of the replica method it has been applied to a wide range of problems within and outside of physics [66, 77, 101]. The most important of these in regards to this thesis is the application of the replica method to calculating the eigenvalues of an Erdős-Rényi random graph [39] (see section 2.4) by Kühn et al. [66] and the application to calculating the learning curves of a GP for Euclidean input spaces by Malzahn and Opper [72].

The idea behind the replica method is simple: since the average of the logarithm of a function is hard to calculate, we replace this average by the easier to calculate logarithm of an average of integer n-replicated versions of the function, and perform a continuation $n \to 0$ at the end. For a function $\alpha(G)$ that depends on a graph drawn from an ensemble

$\mathcal{G}$ we have

$$\begin{aligned}\langle \log \alpha(G) \rangle_{\mathcal{G}} &= \left\langle \lim_{n\to 0} \frac{1}{n} \log \alpha(G)^n \right\rangle_{\mathcal{G}} = \left\langle \lim_{n\to 0} \frac{1}{n} (\alpha(G)^n - 1) \right\rangle_{\mathcal{G}} \\ &= \lim_{n\to 0} \frac{1}{n} \log \langle \alpha(G)^n \rangle_{\mathcal{G}}.\end{aligned} \tag{2.2.1}$$

In most cases in this thesis $\alpha(G)$ will itself be an average of a function $\beta(\xi; G)$. The replica method will therefore be employed in conjunction with a *generating partition function*. In this case one introduces a partition function $Z(\lambda) = \int \mathrm{d}\xi \, \exp(-H(\xi; G) - \lambda\beta(\xi; G))$ that acts as a normaliser to a distribution $P(\xi) \propto \exp(-H(\xi; G))$ for λ equal to zero. In this situation $Z(\lambda)$ depends on the 'disorder' G. Using the partition function, averages with respect to P and subsequently the disorder can be calculated via

$$\begin{aligned}\left\langle \int \mathrm{d}\xi \, \frac{\beta(\xi; G)}{Z(0)} \exp(-H(\xi; G)) \right\rangle_{\mathcal{G}} &= -\lim_{\lambda\to 0} \frac{\partial}{\partial \lambda} \Big\langle \log Z(\lambda) \Big\rangle_{\mathcal{G}} \\ &= -\lim_{\lambda\to 0} \lim_{n\to 0} \frac{\partial}{\partial \lambda} \frac{1}{n} \log \langle Z(\lambda)^n \rangle_{\mathcal{G}}\end{aligned} \tag{2.2.2}$$

Since the replica method is best understood by example, we will now study the application of the replica method for approximating the generalisation error of GP regression derived in Malzahn and Opper [72]. The result obtained can be interpreted as a more general version of the learning curve approximation presented in Sollich [112].

2.2.1 The replica method for learning curves

In Malzahn and Opper [72] the objective was to approximate the generalisation error (2.1.45) with a squared loss function of a Bayesian learner using the replica method. The paper then specialises to GP regression. I will detail here the derivation of the learning curve (2.1.46) for a GP with a Gaussian likelihood. This derivation will serve both as an instructive introduction to the replica method and as a useful baseline for analysing later results.

We begin first by constructing a *generating function* which we can then use to calculate the relevant averages required to compute the generalisation error. We define a partition function of a posterior GP that has seen N examples which have been generated

according to some distribution $p(\boldsymbol{x}, y) = p(\boldsymbol{x})p(\boldsymbol{y}|\boldsymbol{x})$ by

$$Z_N = \int \mathrm{d}f P(f) \exp\left(-\frac{1}{2\sigma^2}\sum_{\mu=1}^{N}(f(\boldsymbol{x}_\mu) - y_\mu)^2\right) \tag{2.2.3}$$

with the GP prior over functions f represented by $P(f)$. From (2.2.3) we can then construct a 'free energy' via

$$F = -\langle\langle \log Z_N \rangle_{\boldsymbol{y}|\boldsymbol{X}}\rangle_{\boldsymbol{X}} = \lim_{n\to 0}\frac{1}{n}\log\langle\langle Z_N^n\rangle_{\boldsymbol{y}|\boldsymbol{X}}\rangle_{\boldsymbol{X}}. \tag{2.2.4}$$

We will show that from this free energy, using (2.2.2), we will be able to construct the required averages to calculate the learning curve.

The replicated average of (2.2.3) in (2.2.4) for most interesting cases is not analytically tractable so an approximation is required. To this end we begin by constructing a *grand canonical form* by considering the case where the number of examples is not fixed but instead fluctuating according to a Poisson distribution with mean ζ

$$\Xi_n(\zeta) = \sum_{N=0}^{\infty}\frac{\zeta^N}{N!}\langle\langle Z_N^n\rangle_{\boldsymbol{y}|\boldsymbol{X}}\rangle_{\boldsymbol{X}}. \tag{2.2.5}$$

(It is worth noting here the $\exp(-\zeta)$ has been dropped since it is a constant that will drop out in subsequent calculations.)

With (2.2.3) in the grand canonical form we can now rewrite (2.2.5) to have a purely local *Hamiltonian* $H(\{f_a\}, \zeta)$ (where there are no interactions between examples $(\boldsymbol{x}_\mu, y_\mu)$ leaving only a local field) involving averages over a single input $\boldsymbol{x}$ and corresponding output y. Since $(\boldsymbol{x}_\mu, y_\mu)$ pairs are i.i.d. we have

$$\begin{aligned}\Xi_n(\zeta) = \int \prod_{a=1}^{n}\mathrm{d}f_a \prod_{a=1}^{n} P(f_a) \sum_{N=0}^{\infty}\frac{\zeta^N}{N!} \\ \times\left(\left\langle\left\langle \exp\left(-\frac{1}{2\sigma^2}\sum_{a=1}^{n}(f_a(\boldsymbol{x}) - y)^2\right)\right\rangle_{y|\boldsymbol{x}}\right\rangle_{\boldsymbol{x}}\right)^N.\end{aligned} \tag{2.2.6}$$

The internal sum over N in (2.2.6) is just a power series of an exponential. Utilising this in (2.2.6) gives us

$$\Xi_n(\zeta) = \int \prod_{a=1}^{n}\mathrm{d}f_a \prod_{a=1}^{n} P(f_a)\exp\big[-H(\{f_a\}, \zeta)\big] \tag{2.2.7}$$

$$H(\{f_a\}, \zeta) = \langle \mathcal{H}(\{f_a\}, \zeta, \boldsymbol{x})\rangle_{\boldsymbol{x}} \tag{2.2.8}$$

$$\mathcal{H}(\{f_a\}, \zeta, \boldsymbol{x}) = -\zeta\left\langle \exp\left(-\frac{1}{2\sigma^2}\sum_{a=1}^{n}(f_a(\boldsymbol{x}) - y)^2\right)\right\rangle_{y|\boldsymbol{x}} \tag{2.2.9}$$

In general (2.2.7) is not solvable analytically. Malzahn and Opper [72] proceed by using a Gaussian variational approximation [42]. They choose to approximate the true Hamiltonian (2.2.8) with a Hamiltonian of quadratic form

$$H_0(\{f_a\}) = \langle \mathcal{H}_0(\{f_a\}, \boldsymbol{x})\rangle_{\boldsymbol{x}} \tag{2.2.10}$$

$$\mathcal{H}_0(\{f_a\}, \boldsymbol{x}) = \sum_{a\leq b} \hat{Q}_{ab}(\boldsymbol{x}) f_a(\boldsymbol{x}) f_b(\boldsymbol{x}) + \sum_{a=1}^{n} \hat{R}_a(\boldsymbol{x}) f_a(\boldsymbol{x}) \tag{2.2.11}$$

so that the difference between the free energies is minimised. This is achieved by finding a H_0 that minimises the Bogoliubov bound

$$-\log \Xi_n(\zeta) \leq -\log \int \prod_{a=1}^{n} \mathrm{d}f_a \prod_{a=1}^{n} P(f_a) e^{-H_0} + \Big\langle H - H_0 \Big\rangle_0. \tag{2.2.12}$$

where $\langle \cdot \rangle_0$ denotes an average with respect to the distribution $\prod_{a=1}^{n} P(f_a) e^{-H_0}$. The RHS of (2.2.12) is known as the *variational free energy*.

Variational minimisation of the variational free energy with respect to $\hat{R}_a(\boldsymbol{x})$ and $\hat{Q}_{ab}(\boldsymbol{x})$ gives the conditions

$$\frac{\delta \langle H - H_0\rangle_0}{\delta \hat{R}_a(\boldsymbol{x})} = 0 \quad \forall a \tag{2.2.13}$$

$$\frac{\delta \langle H - H_0\rangle_0}{\delta \hat{Q}_{ab}(\boldsymbol{x})} = 0 \quad \forall a \leq b \tag{2.2.14}$$

where the functional derivative acts only on the averages and not on the internal $H - H_0$ term.

Taking a second, closer look, at the variational free energy one sees that, since H_0 is quadratic in form, the RHS of (2.2.12) will be completely determined by the first two local moments $R_a(\boldsymbol{x}) = \langle f_a(\boldsymbol{x})\rangle_0$ and $Q_{ab}(\boldsymbol{x}, \boldsymbol{x}) = \langle f_a(\boldsymbol{x}) f_b(\boldsymbol{x})\rangle_0$. With this realisation we can derive a second set of variational equations which must be satisfied by performing a Legendre transform upon the variational free energy. We rewrite the free energy for the Hamiltonian H_0 given by $F_0 = -\log \int \prod_a \mathrm{d}f_a \prod_{a=1}^{n} P(f_a) e^{-H_0}$ in terms of its Legendre transform, denoted G_0, to get

$$\begin{aligned} G_0 &= F_0 - \int \mathrm{d}\boldsymbol{x} \sum_a \hat{R}_a(\boldsymbol{x}) \frac{\partial F_0}{\partial \hat{R}_a(\boldsymbol{x})} - \int \mathrm{d}\boldsymbol{x} \sum_{a\leq b} \hat{Q}_{ab}(x) \frac{\partial F_0}{\partial \hat{Q}_{ab}(\boldsymbol{x})} \\ &= F_0 - \int \mathrm{d}\boldsymbol{x} \sum_a \hat{R}_a(\boldsymbol{x}) p(\boldsymbol{x}) \langle f_a(\boldsymbol{x})\rangle_0 - \int \mathrm{d}\boldsymbol{x} \sum_{a\leq b} \hat{Q}_{ab}(\boldsymbol{x}) p(\boldsymbol{x}) \langle f_a(\boldsymbol{x}) f_b(\boldsymbol{x})\rangle_0 \\ &= F_0 - \left\langle \sum_a \hat{R}_a(\boldsymbol{x}) R_a(\boldsymbol{x}) \right\rangle_{\boldsymbol{x}} - \left\langle \sum_{a\leq b} \hat{Q}_{ab}(\boldsymbol{x}) Q_{ab}(\boldsymbol{x}, \boldsymbol{x}) \right\rangle_{\boldsymbol{x}} = F_0 - \langle H_0\rangle_0 \end{aligned} \tag{2.2.15}$$

Differentiating with respect to $R_c(\boldsymbol{x})$ we see that we have

$$\begin{aligned}\frac{\delta G_0}{\delta R_c(\boldsymbol{x})} &= \int \mathrm{d}\boldsymbol{x}' \sum_a \frac{\delta F_0}{\delta \hat{R}_a(\boldsymbol{x}')}\frac{\delta \hat{R}_a(\boldsymbol{x}')}{\delta R_c(\boldsymbol{x})} + \int \mathrm{d}\boldsymbol{x}' \sum_{a\leq b} \frac{\delta F_0}{\delta \hat{Q}_{ab}(\boldsymbol{x}')}\frac{\delta \hat{Q}_{ab}(\boldsymbol{x}')}{\delta R_c(\boldsymbol{x})} \\ &\quad - \hat{R}_c(\boldsymbol{x})p(\boldsymbol{x}) - \left\langle \sum_a \frac{\delta \hat{R}_a(\boldsymbol{x}')}{\delta R_c(\boldsymbol{x})} R_a(\boldsymbol{x}') \right\rangle_{\boldsymbol{x}'} - \left\langle \sum_{a\leq b} \frac{\delta \hat{Q}_{ab}(\boldsymbol{x}')}{\delta R_c(\boldsymbol{x})} Q_{ab}(\boldsymbol{x}', \boldsymbol{x}') \right\rangle_{\boldsymbol{x}'} \\ &= -\hat{R}_c(\boldsymbol{x})p(\boldsymbol{x}) \end{aligned} \tag{2.2.16}$$

Similar arguments for $Q_{ab}(\boldsymbol{x}, \boldsymbol{x})$ give

$$\frac{\delta G_0}{\delta Q_{ab}(\boldsymbol{x}, \boldsymbol{x})} = -p(\boldsymbol{x})\hat{Q}_{ab}(\boldsymbol{x}, \boldsymbol{x}) \tag{2.2.17}$$

If we then substitute the definition of G_0 into the variational free energy (2.2.12) and differentiate with respect to $R_a(\boldsymbol{x})$ we get

$$\begin{aligned} 0 = \frac{\delta F_0}{\delta R_a(\boldsymbol{x})} + \frac{\delta \langle H - H_0 \rangle_0}{\delta R_a(\boldsymbol{x})} &= \frac{\delta G_0}{\delta R_a(\boldsymbol{x})} + \frac{\delta \langle H \rangle_0}{\delta R_a(\boldsymbol{x})} \\ &= -\hat{R}_a(\boldsymbol{x})p(\boldsymbol{x}) + \frac{\delta \langle H \rangle_0}{\delta R_a(\boldsymbol{x})} \\ &= -\hat{R}_a(\boldsymbol{x})p(\boldsymbol{x}) + p(\boldsymbol{x})\frac{\partial \langle \mathcal{H} \rangle_0}{\partial R_a(\boldsymbol{x})} \end{aligned} \tag{2.2.18}$$

again with an analogous result for $Q_{ab}(\boldsymbol{x}, \boldsymbol{x})$.

From equations (2.2.13), (2.2.14) and (2.2.18) we now have a set of variational equations that we must satisfy for H_0 to minimise (2.2.12). To summarise, we aim to minimise the variational free energy (2.2.12) and we have calculated that at a minimum the following conditions must be met

$$\frac{\delta \langle H - H_0 \rangle_0}{\delta \hat{R}_a(\boldsymbol{x})} = 0 \quad \forall a, \qquad \frac{\delta \langle H - H_0 \rangle_0}{\delta \hat{Q}_{ab}(\boldsymbol{x})} = 0 \quad a \leq b, \tag{2.2.19}$$

$$\frac{\partial \langle \mathcal{H} \rangle_0}{\partial R_a(\boldsymbol{x})} = \hat{R}_a(\boldsymbol{x}) \quad \forall a, \qquad \frac{\partial \langle \mathcal{H} \rangle_0}{\partial Q_{ab}(\boldsymbol{x}, \boldsymbol{x})} = \hat{Q}_{ab}(\boldsymbol{x}) \quad a \leq b. \tag{2.2.20}$$

To minimise (2.2.12) we will make the simplest assumption that satisfies the above constraints, namely *replica symmetry*

$$\hat{R}_a(\boldsymbol{x}) = \hat{R}(\boldsymbol{x}) \quad \forall a, \qquad \hat{Q}_{aa}(\boldsymbol{x}) = \frac{1}{2}\hat{Q}_0(\boldsymbol{x}) \quad \forall a, \qquad \hat{Q}_{ab}(\boldsymbol{x}) = \hat{Q}(\boldsymbol{x}) \quad a < b. \tag{2.2.21}$$

This assumed replica symmetric nature of the variational parameters then enforces this symmetry upon the first two moments, i.e. the first two derivatives of the variational

free energy giving

$$\langle f_a(\boldsymbol{x})\rangle_0 = R(\boldsymbol{x}) \quad \forall a, \tag{2.2.22}$$

$$\langle f_a(\boldsymbol{x}) f_a(\boldsymbol{x}')\rangle_0 = Q_0(\boldsymbol{x}, \boldsymbol{x}') \quad \forall a, \tag{2.2.23}$$

$$\langle f_a(\boldsymbol{x}) f_b(\boldsymbol{x}')\rangle_0 = Q(\boldsymbol{x}, \boldsymbol{x}') \quad \forall a < b. \tag{2.2.24}$$

One can show, by arguing in reverse, the meaning of $R(\boldsymbol{x})$, $Q(\boldsymbol{x}, \boldsymbol{x}')$ and $Q_0(\boldsymbol{x}, \boldsymbol{x}')$. We see that

$$\begin{aligned}
&\langle\langle\langle f(\boldsymbol{x})\rangle_f\rangle_{\boldsymbol{y}|\boldsymbol{X}}\rangle_{\boldsymbol{X}} \\
&\quad = \left\langle \left\langle \frac{1}{Z_N} \int \mathrm{d}f\, P(f) f(\boldsymbol{x}) \exp\left(-\frac{1}{2\sigma^2} \sum_{\mu=1}^{N} (f(\boldsymbol{x}_\mu) - y_\mu)^2 \right) \right\rangle_{\boldsymbol{y}|\boldsymbol{X}} \right\rangle_{\boldsymbol{X}} \\
&\quad = \lim_{n\to 0} \left\langle \left\langle Z_N^{n-1} \int \mathrm{d}f\, P(f) f(\boldsymbol{x}) \exp\left(-\frac{1}{2\sigma^2} \sum_{\mu=1}^{N} (f(\boldsymbol{x}_\mu) - y_\mu)^2 \right) \right\rangle_{\boldsymbol{y}|\boldsymbol{X}} \right\rangle_{\boldsymbol{X}} \\
&\quad \approx \lim_{n\to 0} \int \prod_a \mathrm{d}f_a \prod_a P(f_a) f_1(\boldsymbol{x}) \exp\left(-H\right) \\
&\quad \approx \lim_{n\to 0} \int \prod_a \mathrm{d}f_a \prod_a P(f_a) f_1(\boldsymbol{x}) \exp\left(-H_0\right) \\
&\quad = R(\boldsymbol{x})
\end{aligned} \tag{2.2.25}$$

which is just the posterior mean and where we have taken $\zeta = N$. Similar calculations for $Q_0(\boldsymbol{x}, \boldsymbol{x}')$ and $Q(\boldsymbol{x}, \boldsymbol{x}')$ show that $Q_0(\boldsymbol{x}, \boldsymbol{x}') \approx \langle\langle\langle f(\boldsymbol{x}) f(\boldsymbol{x}')\rangle_f\rangle_{\boldsymbol{y}|\boldsymbol{X}}\rangle_{\boldsymbol{X}}$ and $Q(\boldsymbol{x}, \boldsymbol{x}') \approx \langle\langle\langle f(\boldsymbol{x})\rangle_f \langle f(\boldsymbol{x}')\rangle_f\rangle_{\boldsymbol{y}|\boldsymbol{X}}\rangle_{\boldsymbol{X}}$.

Since we have made a Gaussian approximation, (2.2.8) describes a Gaussian process that *includes the data average*. Thus the variational approximation, H_0, of the true Hamiltonian, H, describes a GP with both the prior and data averages included. We define

$$G(\boldsymbol{x}, \boldsymbol{x}') = Q_0(\boldsymbol{x}, \boldsymbol{x}') - Q(\boldsymbol{x}, \boldsymbol{x}') \tag{2.2.26}$$

to be the average posterior covariance function.

Closer inspection of the distribution averaged over in (2.2.22) to (2.2.24) shows that since the prior, $P(f)$, was a GP, the distribution induced by H_0 is a GP with an exponent

which may be written as

$$-\frac{1}{2}\sum_a \int \mathrm{d}\boldsymbol{x}\mathrm{d}\boldsymbol{x}'\, f_a(\boldsymbol{x})C^{-1}(\boldsymbol{x},\boldsymbol{x}')f_a(\boldsymbol{x}') \\ -\frac{1}{2}\sum_{a,b}\int \mathrm{d}\boldsymbol{x}\, p(\boldsymbol{x})f_a(\boldsymbol{x})(\hat{Q}_0(\boldsymbol{x})\delta_{ab}+\hat{Q}(\boldsymbol{x})(1-\delta_{ab}))f_b(\boldsymbol{x}) \\ -\sum_a \int \mathrm{d}\boldsymbol{x}\, p(\boldsymbol{x})\hat{R}(\boldsymbol{x})f_a(\boldsymbol{x}) \quad (2.2.27)$$

where C^{-1} is the operator inverse of C. If we also define the operator $\hat{Q}_0$ as $\hat{Q}_0(\boldsymbol{x},\boldsymbol{x}') = p(\boldsymbol{x})\hat{Q}_0(\boldsymbol{x})\delta(\boldsymbol{x}-\boldsymbol{x}')$, and $\hat{Q}$ similarly, then we can further simplify this to read

$$-\frac{1}{2}\sum_{a,b}\int \mathrm{d}\boldsymbol{x}\mathrm{d}\boldsymbol{x}'\, f_a(\boldsymbol{x})((C^{-1}+\hat{Q}_0)\delta_{ab}+\hat{Q}(1-\delta_{ab}))(\boldsymbol{x},\boldsymbol{x}')f_b(\boldsymbol{x}') \\ -\sum_a \int \mathrm{d}\boldsymbol{x}\, p(\boldsymbol{x})\hat{R}(\boldsymbol{x})f_a(\boldsymbol{x}) \quad (2.2.28)$$

In order to calculate the mean, $R(\boldsymbol{x})$, and covariance function, $D(\boldsymbol{x},\boldsymbol{x}')$, for the GP described by the distribution induced by H_0 we therefore only need to complete the square above and write the exponent in the form

$$-\frac{1}{2}\sum_{a,b}\int \mathrm{d}\boldsymbol{x}\mathrm{d}\boldsymbol{x}'\,(f_a(\boldsymbol{x})-R(\boldsymbol{x}))D^{-1}_{ab}(\boldsymbol{x},\boldsymbol{x}')(f_b(\boldsymbol{x}')-R(\boldsymbol{x}')) \quad (2.2.29)$$

One finds that the solution for D is given by

$$D_{ab}(\boldsymbol{x},\boldsymbol{x}') = (C^{-1}+u)^{-1}(\boldsymbol{x},\boldsymbol{x}')\delta_{ab} \\ -\int \mathrm{d}\boldsymbol{x}''(C^{-1}+u_{n-1})^{-1}(\boldsymbol{x},\boldsymbol{x}'')\left[p(\boldsymbol{x}'')\hat{Q}(\boldsymbol{x}'')\right](C^{-1}+u)^{-1}(\boldsymbol{x}'',\boldsymbol{x}') \quad (2.2.30)$$

and $R(\boldsymbol{x})$ is

$$R(\boldsymbol{x}) = -\int \mathrm{d}\boldsymbol{x}'\Bigg[(C^{-1}+u)^{-1}(\boldsymbol{x},\boldsymbol{x}') \\ -n\int \mathrm{d}\boldsymbol{x}''\,(C^{-1}+u_{n-1})^{-1}(\boldsymbol{x},\boldsymbol{x}'')\left[p(\boldsymbol{x}'')\hat{Q}(\boldsymbol{x}'')\right](C^{-1}+u)^{-1}(\boldsymbol{x}'',\boldsymbol{x}')\Bigg]p(\boldsymbol{x}')\hat{R}(\boldsymbol{x}') \quad (2.2.31)$$

with

$$u(\boldsymbol{x},\boldsymbol{x}') = p(\boldsymbol{x})(\hat{Q}_0(\boldsymbol{x})-\hat{Q}(\boldsymbol{x}))\delta(\boldsymbol{x}-\boldsymbol{x}') \quad (2.2.32)$$

$$u_{n-1}(\boldsymbol{x},\boldsymbol{x}') = p(\boldsymbol{x})(\hat{Q}_0(\boldsymbol{x})+(n-1)\hat{Q}(\boldsymbol{x}))\delta(\boldsymbol{x}-\boldsymbol{x}') \quad (2.2.33)$$

and where inverses are taken to mean the inverse operator.

From these equations we see that in the limit as n tends to zero the approximation of the posterior covariance of (2.1.49) will be given by

$$\begin{aligned} G(\boldsymbol{x}, \boldsymbol{x}') &= Q_0(\boldsymbol{x}, \boldsymbol{x}') - Q(\boldsymbol{x}, \boldsymbol{x}') = D_{aa}(\boldsymbol{x}, \boldsymbol{x}') - D_{ab}(\boldsymbol{x}, \boldsymbol{x}') \qquad a \neq b \\ &= (C^{-1} + u)^{-1}(\boldsymbol{x}, \boldsymbol{x}'), \end{aligned} \tag{2.2.34}$$

and the posterior mean will be

$$R(\boldsymbol{x}) = -\langle G(\boldsymbol{x}, \boldsymbol{x}')\hat{R}(\boldsymbol{x}')\rangle_{\boldsymbol{x}'} \tag{2.2.35}$$

Combining conditions (2.2.20) with definition (2.2.34) we see that (2.2.20) becomes [72]

$$\hat{Q}_0(\boldsymbol{x}) - \hat{Q}(\boldsymbol{x}) = \frac{N}{\sigma^2 + G(\boldsymbol{x}, \boldsymbol{x})}. \tag{2.2.36}$$

Subsequently combining (2.2.34) with (2.2.36) we have a self-consistent equation for the approximate posterior covariance, G. Note that this distribution is agnostic to the outputs $\boldsymbol{y}$. This is to be expected since, as we saw in (2.1.9), the posterior covariance of a GP is determined only by the input points $\boldsymbol{x}$ and not the output points, $\boldsymbol{y}$.

We are now in a position to calculate the matched learning curve (2.1.47) for the case of inputs and outputs defined on the vertices of a graph. From (2.1.47) we see that for the case of a matched model, this will be given by the posterior covariance of the GP averaged over the graph ensemble $\mathcal{G}$ and inputs $\boldsymbol{x}$. In terms of the replica calculation this is simply the average of $G(\boldsymbol{x}, \boldsymbol{x})$ over the input distribution of $\boldsymbol{x}$ and graph ensemble $\mathcal{G}$.

For an input distribution $p(\boldsymbol{x})$ we have,

$$\epsilon_g = \left\langle \int \mathrm{d}\boldsymbol{x}\, p(\boldsymbol{x}) G(\boldsymbol{x}, \boldsymbol{x}) \right\rangle_{\mathcal{G}} = \left\langle \int \mathrm{d}\boldsymbol{x}\, p(\boldsymbol{x}) (C^{-1} + u)^{-1}(\boldsymbol{x}, \boldsymbol{x}) \right\rangle_{\mathcal{G}} \tag{2.2.37}$$

If we assume, as in Sollich [112], that we may replace G by its average ϵ_g in the denominator of (2.2.36) then we have the approximation

$$\epsilon_g = \left\langle \int \mathrm{d}\boldsymbol{x}\, p(\boldsymbol{x}) \left(C^{-1} + \frac{N}{\sigma^2 + \epsilon_g} \right)^{-1} (\boldsymbol{x}, \boldsymbol{x}) \right\rangle_{\mathcal{G}}. \tag{2.2.38}$$

One can simplify this further by defining a positive semi-definite operator $P^{1/2}CP^{1/2}(\boldsymbol{x}, \boldsymbol{x}') = p(\boldsymbol{x})^{1/2} C(\boldsymbol{x}, \boldsymbol{x}') p^{1/2}(\boldsymbol{x}')$ and then use Mercer's theorem [135] to decompose $P^{1/2}CP^{1/2}$

into its normalised eigenfunctions $\psi_i(\boldsymbol{x})$ and eigenvalues λ_i so that $P^{1/2}CP^{1/2}(\boldsymbol{x},\boldsymbol{x}') = \sum_{i=1}^{\infty}\lambda_i\psi_i(\boldsymbol{x})\psi_i(\boldsymbol{x}')$ with $\int \mathrm{d}\boldsymbol{x}\,\psi_i(\boldsymbol{x})\psi_j(\boldsymbol{x}) = \delta_{ij}$. From this we see that by defining $p^{1/2}(\boldsymbol{x})\phi(\boldsymbol{x}) = \psi(\boldsymbol{x})$, that $C(\boldsymbol{x},\boldsymbol{x}') = \sum_{i=1}^{\infty}\lambda_i\phi_i(\boldsymbol{x})\phi_i(\boldsymbol{x}')$ with $\int \mathrm{d}\boldsymbol{x}\,p(\boldsymbol{x})\phi_i(\boldsymbol{x})\phi_j(\boldsymbol{x}) = \delta_{ij}$. Substituting this eigenfunction form of C into (2.2.38) we have

$$\begin{aligned}
&\left\langle \int \mathrm{d}\boldsymbol{x}\,p(\boldsymbol{x})\left(C^{-1}+\frac{N}{\sigma^2+\epsilon_g}\right)^{-1}(\boldsymbol{x},\boldsymbol{x})\right\rangle_{\mathcal{G}} \\
&\qquad = \left\langle \int \mathrm{d}\boldsymbol{x}\,p(\boldsymbol{x})\sum_{i=1}^{\infty}\phi_i(\boldsymbol{x})\left(\lambda_i^{-1}+\frac{N}{\sigma^2+\epsilon_g}\right)^{-1}\phi_i(\boldsymbol{x})\right\rangle_{\mathcal{G}} \\
&\qquad = \left\langle \sum_{i=1}^{\infty}\left(\lambda_i^{-1}+\frac{N}{\sigma^2+\epsilon_g}\right)^{-1}\int \mathrm{d}\boldsymbol{x}\,p(\boldsymbol{x})\phi_i(\boldsymbol{x})\phi_i(\boldsymbol{x})\right\rangle_{\mathcal{G}} \\
&\qquad = \left\langle \sum_{i=1}^{\infty}\left(\lambda_i^{-1}+\frac{N}{\sigma^2+\epsilon_g}\right)^{-1}\right\rangle_{\mathcal{G}}
\end{aligned} \tag{2.2.39}$$

thus we arrive at the learning curve approximation called ϵ_{LC} in Sollich [112]

$$\epsilon_g = \left\langle g\left(\frac{N}{\sigma^2+\epsilon_g}\right)\right\rangle_{\mathcal{G}}, \qquad g(h) = \sum_{i=1}^{\infty}(\lambda_i^{-1}+h)^{-1} \tag{2.2.40}$$

We will use (2.2.40) as a baseline comparison for evaluating the performance of the approximations we will derive in subsequent chapters. It is worth noting that since the random walk kernel is constructed from the normalised Laplacian, and for some of the random graphs we will consider in the remainder of this thesis the eigenvalue distribution approaches a limiting form for large V [75] we will be able to simplify this further. In these situations we may evaluate (2.2.40) using the limiting form and drop the average over the ensemble.

2.3 Cavity method and belief propagation – Two sides of the same coin

When Sherrington and Kirkpatrick [106] solved the SK model using the replica method they found that, as temperatures became small, predictions for the free energy became inaccurate. To understand these inaccuracies for small temperatures an alternative, cavity approach [86], was proposed by Thouless et al. [122]. The resulting equations were called the *Thouless-Anderson-Palmer* (TAP) equations and led to the eventual discovery of replica symmetry breaking [89–91].

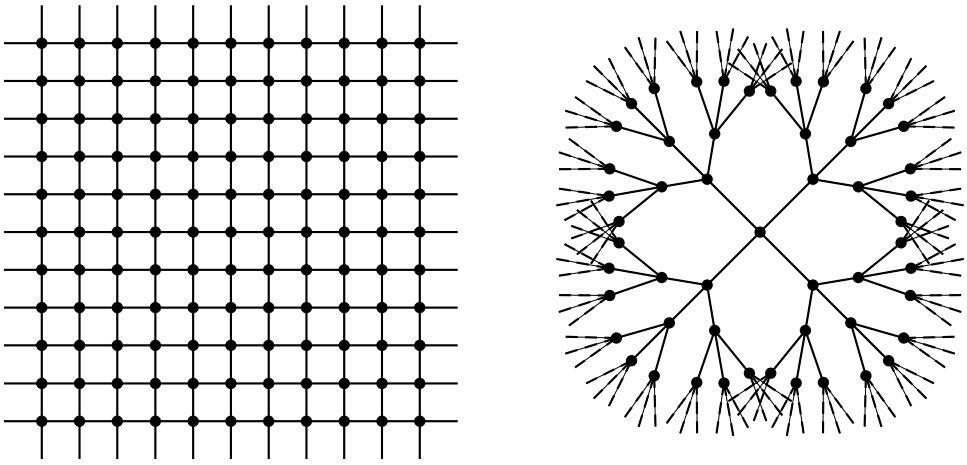


Figure 2.5: (left) The EA model of a spin glass, spins are placed on a regular lattice and interact with immediate neighbours only. (right) The VB model of a spin glass, spins interact with a fixed number of randomly chosen spins or neighbours.

The difference, in essence, between the replica solution to the free energy and the TAP equations is the point at which one makes a mean field approximation: in replicas this is done after averaging out disorder and in the TAP equations it is taken before averaging out disorder.

The cavity derived TAP equations were first calculated for the SK model. However, although the SK model was found to exhibit some of the properties of real world spin glass systems, it made unrealistic assumptions about the interactions between spins (namely that every spin interacted with every other spin). As a compromise between the unsolvable EA model (where spins are placed on a lattice and interactions only occur between finitely many neighbours) and the overly simple SK model, Viana and Bray [126] proposed the *Viana-Bray* (VB) model (see figure 2.5). In this model spins interacted with finitely many other spins like the EA model whilst preserving the typically long cycles in the SK model that made it solvable. It is on a less restrictive version of the original VB model that we will derive the *cavity method* as it most closely resembles the problems we will solve in subsequent chapters. To tie in with later derivations we will derive the cavity method in a similar manner to that of Yedidia et al. [136]: in the setting of distributions placed over a random graph.

Since the development of the cavity method it has been found that it has been invented in

other subject areas under various different names. In particular, it was created as a way of approximating distributions defined on graphical models [20] in computer science [93]. In this field it is known as *belief propagation* (BP). In the field of coding theory it was created for decoding transmitted information [45] and was called the *sum-product algorithm*.

We begin our derivation by considering a fixed graph $G(\mathcal{V}, \mathcal{E})$ with vertex set $\mathcal{V} = \{1, \ldots, V\}$ and edge set $\mathcal{E} = \{(i,j)|i \text{ connected to } j\}$ and assume we have a function $\boldsymbol{f} = (f(1), \ldots, f(V))^{\mathrm{T}}$ defined on the vertices of the graph with joint probability distribution of the form

$$P(f_1, \ldots, f_V) = \prod_{(ij)\in\mathcal{E}} \psi_{ij}(f_i, f_j) \prod_{i\in\mathcal{V}} \phi_i(f_i). \tag{2.3.1}$$

In the machine learning literature this is known as an *undirected graphical model* or a *Markov random field* [61]. We will suppose that we require the marginal distribution for f_i. Integrating over all components of $\boldsymbol{f}$ except i, which we will denote by $\boldsymbol{f}_{\setminus i}$ then gives

$$P(f_i) \propto \phi_i(f_i) \int \prod_{j\neq i} \mathrm{d}f_j \prod_{j\in\mathcal{N}(i)} \psi_{ij}(f_i, f_j) P^{(i)}(\boldsymbol{f}_{\setminus i}) \tag{2.3.2}$$

where we have denoted the neighbourhood of vertex i by $\mathcal{N}(i)$ and

$$P^{(i)}(\boldsymbol{f}_{\setminus i}) \propto \prod_{k\in\mathcal{V}\setminus\{i\}} \phi_j(f_j) \prod_{\substack{(j,k)\in \\ \mathcal{E}\setminus\{(i,j)|j\in\mathcal{N}(i)\}}} \psi_{j,k}(f_j, f_k). \tag{2.3.3}$$

Equation (2.3.3) represents a cavity field [86]; in statistical mechanics this can be interpreted as the distribution of the system in the absence of site or vertex i. Equation (2.3.2) can be interpreted as the effect of adding an additional site i to the system.

If we assume the graph is 'tree-like' i.e. that cycles in G are large and grow with system size, and we assume that correlations between vertices arising from such cycles are not too strong, then interactions between vertices can be approximated by considering the shortest paths between them. Vertices with shortest paths passing through i will become approximately independent of each other upon the removal of i. We therefore approximate the graph G without vertex i by independent subgraphs rooted at the neighbours of i (see figure 2.6).

Using the independence of the subgraphs after removing vertex i, we now have the

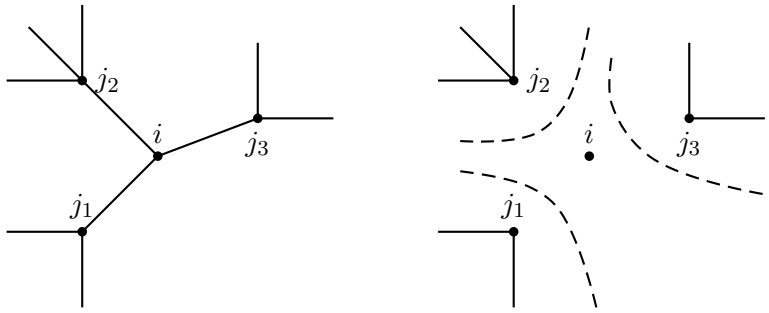


Figure 2.6: Figure to explain the cavity method. By removing vertex i neighbouring vertices can be approximated as roots of independent subgraphs.

iterative equation

$$P_j^{(i)}(f_j) \propto \phi_j(f_j) \int \prod_{k \in \mathcal{N}(j)\setminus\{i\}} \mathrm{d}f_k \, \psi_{jk}(f_j, f_k) P_k^{(j)}(f_k). \tag{2.3.4}$$

If G was a tree eventually we must reach the leaves of the tree; these will simply have cavity distributions $P_k^{(j)}(f_k) = P_k(f_k)$. Working backwards would give the marginal (2.3.2). If G is not a tree, however, then there will be no leaves to terminate this iteration. As an approximation, one might be tempted to try randomly assigning all cavity marginals in the graph an initial value. One could then update the cavity messages at random until convergence. This approximate method is known as *belief propagation* (BP).

Equations (2.3.2) and (2.3.4) solve the problem of calculating marginals of a distribution on a fixed random graph. To gain a better insight into these equations we see that the integral terms

$$m_{i \leftarrow j}(f_i) = \int \mathrm{d}f_j \psi_{ij}(f_i, f_j) P_j^{(i)}(f_j) \tag{2.3.5}$$

in (2.3.2) and (2.3.4) can be interpreted as the message sent from a neighbouring vertex j to i about its 'belief' in what value f_i should be. This interpretation of the update step along with an interpretation of the final marginal calculation is summarised in figure 2.7.

If all we needed was to perform inference on fixed graphs we would be done. The cavity method, however, was developed to solve the problem of predicting marginal

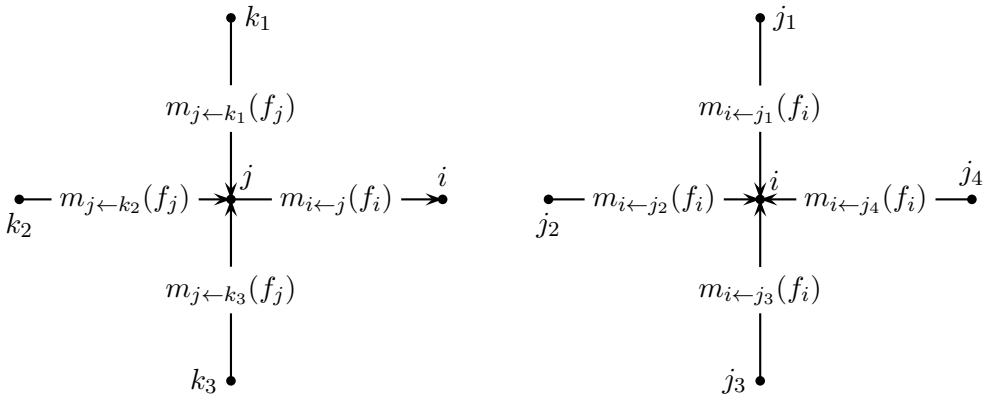


Figure 2.7: Figure to explain the message passing interpretation of the cavity method. (left) A typical update step. (right) A final marginal calculation

distributions of spins in the presence of disorder, which in this case is represented by an ensemble of random graphs. If we consider an ensemble of random graphs constrained only by a degree distribution (the distribution of the degree of a randomly selected vertex, see section 2.4) with finite mean, the cycle length of a typical graph will be $O(\log(V))$. As the size of the graph becomes large ($V \to \infty$), cycles are rare, and we may once again make the independent subgraph approximation used to calculate (2.3.4). In this case the assumption translates into assuming that incoming messages are independent of one another. Equation (2.3.4) therefore becomes a self consistent equation relating the probability of a belief $m_{i\leftarrow j}(f)$. We have

$$\pi[m_{i\leftarrow j}] = \sum_d \frac{p(d)d}{\bar{d}} \int \{\mathcal{D}\pi[m_{j\leftarrow k}]\}^{d-1} \delta\left(m_{i\leftarrow j}(f) - \int \mathrm{d}f P_i(f) \prod_{k=1}^{d-1} m_{j\leftarrow k}(f) \right) \quad (2.3.6)$$

which must be solved using population dynamics [2, 78] (see algorithm 4.1). The average over the distribution $p(d)d/\bar{d}$ can be interpreted as an average over edges in the graph; since for a vertex with degree d one expects d edges, $\bar{d}$ normalises the distribution.

The cavity method has had success solving a variety of problems including turbo-codes [15], where it is used to achieve communication at rates close to the Shannon limit, in statistical physics, as an equivalent, more interpretable method for solving spin glass systems [122], and for solving eigenvalue problems in large random graph ensem-

bles [97]. Of particular interest for the rest of this thesis, in particular Chapter 4 and Chapter 6, is the work of Rogers et al. [97] where the authors calculated the eigenvalue spectrum of ensembles large random graphs constrained by a degree distribution $p(d)$.

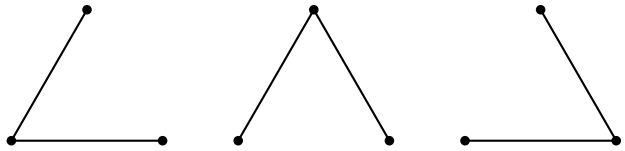
2.4 Random Graphs

Ensembles of random graphs enable one to consider whole classes of graphs that share common properties. As discussed in Chapter 1 many graphs in the real world, from a range of seemingly unrelated areas, share common properties such as power law degree distributions, preferential attachment and community structures. To accurately convey the average performance of GPs for regression on graphs it is desirable therefore to consider the performance of GPs for regression on classes of graphs defined by an ensemble.

Random graphs are a large subject in their own right [9, 18, 23, 26, 29, 82–84, 92, 98, 131]. We will discuss here only the relevant information for this thesis. We will begin by the definition of a random graph ensemble.

Definition 2.4.1 (Random graph ensemble)**.** A random graph ensemble $\mathcal{G}$ of graphs with V vertices is the set of all labelled graphs of size V, $\mathcal{G}^V$, endowed with a probability distribution $P_{\mathcal{G}}$

It is worth noting that the labelled part of the definition is important. It implies that graphs such as



are different. See Bollobás [22] for further discussions on unlabelled graphs.

Subsequent subsections of this overview of random graphs will now detail the ensembles that feature in this thesis. I will highlight important information that is relevant to generating graphs uniformly from the ensemble and discuss, where necessary, how one may sensibly select the parameters of the ensembles. All of the ensembles we discuss can be viewed as a generalisation of the VB spin glass model [126].

2.4.1 Erdős-Rényi random graphs

It is perhaps fitting that random graph theory traces its way back to Paul Erdős [39], the 20th century poly-mathematician that holds the record for the most papers published, and that gave rise to the infamous Erdős number [47] (an integer number that represents the number of coauthors between an author and a paper written by Paul Erdős). It is to the first random graphs considered by Erdős and Rényi [39] that many modern approaches can be traced.

In the seminal papers by Erdős and Rényi [39, 40] the authors considered the following ensemble.

Definition 2.4.2 (Erdős-Rényi (Poisson) random graph ensemble [39])**.** The Erdős-Rényi ensemble $\mathcal{G}_{V,p}$ with $p \in [0,1]$ is defined as the set of all labelled graphs $\mathcal{G}^V$ with probability distribution

$$P_{\mathcal{G}}(G \in \mathcal{G}^V) = p^{|\mathcal{E}|}(1-p)^{V(V-1)/2-|\mathcal{E}|} \tag{2.4.1}$$

Samples from the Erdős-Rényi ensemble can be viewed as being generated by making i.i.d. Bernoulli trials with probability p for each pair in $\{(i,j)|i,j = 1\ldots,V, i < j\}$. If the Bernoulli trial returns one then the edge is present in $\mathcal{E}$ otherwise it is not. This interpretation leads to algorithm 2.1 for generating graphs from the Erdős-Rényi ensemble.

Since the publication of the first basic properties of $\mathcal{G}_{V,p}$ in Erdős and Rényi [39] many authors have studied these graphs and consequently a great deal is known about them [see for example 22]. One of the most useful results as regards to this thesis is the '6 stages' typical samples $G \in \mathcal{G}_{V,p}$ progress through as p is increased from 0 to 1. These results were first realised in Erdős and Rényi [39] but have subsequently been improved upon many times. The description below of the stages is taken from Chung and Lu [26] where the references to the relevant original papers can also be found.

Stage 1 p grows slower than $1/V$: Graphs are *almost surely* (a.s.) a disjoint union of trees.

Stage 2 $p = c/V$, $c \in (0,1)$**:** Graphs contain cycles of any given length. All components are trees or trees with one additional edge (unicyclic components).

Algorithm 2.1: Generating graphs from $\mathcal{G}_{V,p}$

Input : Number of vertices V and the probability of an edge p

Output: Graph $G(\mathcal{V}, \mathcal{E})$

```
1  begin
2      Initialise vertex set V ← {1, ..., V};
3      Initialise edge set E ← ∅;
4      for i ← 0 to V − 1 do
5          for j ← 0 to i − 1 do
6              Sample from a Bernoulli distribution with success probability p;
7              if true then
8                  Add edge (i, j) to E;
9              end
10         end
11     end
12 end
```

Stage 3 $p = 1/V$**:** The largest component has size of order $V^{2/3}$

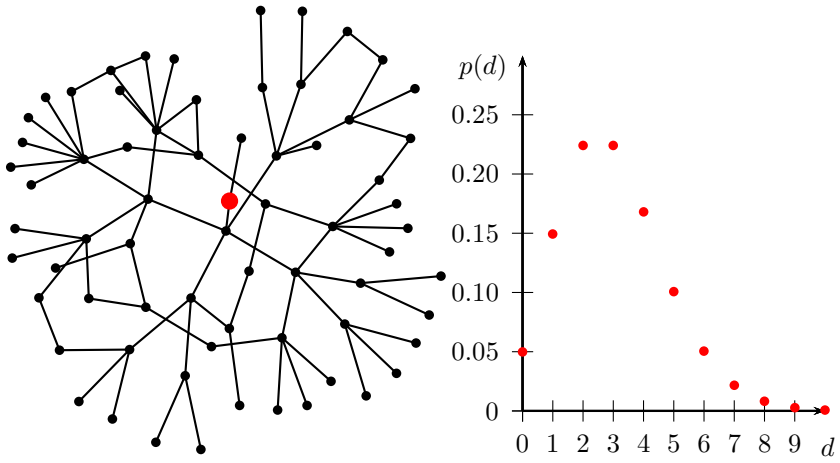
Stage 4 $p = c/V$**,** $c > 1$**:** Except for one giant component all other components are small and most are trees. The total number of vertices in the giant component is approximately $f(c)V$ with

$$f(c) = 1 - \frac{1}{c} \sum_{k=1}^{\infty} \frac{k^{k-1}}{k!} (c\mathrm{e}^{-c})^k. \tag{2.4.2}$$

Stage 5 $p = (c \log V)/V$**,** $c \geq 1$**:** The graph is a.s. connected (there exists a path between any two vertices within the graph)

Stage 6 $p = (\omega(V) \log V)/V$**,** $\omega(V) \to \infty$**:** The graph is a.s. connected and almost all the degrees of the vertices are asymptotically equal.

We focus in this thesis on stage 4. In this stage graphs from the ensemble are complicated enough to make them interesting and mainly connected, whilst preserving the tree-like requirement for the cavity method to be valid. The degree distribution $p(d) = \frac{1}{V} \sum_i \delta_{d_i,d}$, with degree of vertex i given by $d_i = \sum_j a_{ij}$, of this type of graph will be asymptotically



(a) (b)

Figure 2.8: (a) An expansion around a vertex (shown in red) inside the 'giant component' of an Erdős-Rényi random graph with $c = 3$ and five hundred vertices. The tree like stucture is clearly visible. (b) The expected degree distribution of this graph.

Poisson with mean c. A sample of a graph from a stage 4 Erdős-Rényi ensemble with $c = 3$ is given in figure 2.8.

2.4.2 Extending the Erdős-Rényi graph – The generalised random graph ensemble

Until relatively recently the focus of research in the random graph field was upon the Erdős-Rényi ensemble. However, in the early nighties, with the advent of more powerful computers, researchers became able to study in detail the properties of graphs found in nature. The researchers' magnifying glass ranged over a broad spectrum of graphs from the world wide web to phone call graphs to metabolic networks in cells [1, 3, 4, 57]. Studies of these graphs gave what was, at first, surprising results that could not be explained by the typical properties of an Erdős-Rényi graph with edges selected via i.i.d. Bernoulli variables. These were;

Power law (scale free) degree distribution: The degree distribution of these graphs exhibited a power law $p(d) \propto d^{-\gamma}$.

Assortativity/Disassortativity: The degrees of two vertices connected by an edge

are not independent. Graphs either had an assortative property where the degrees of connected vertices were positively correlated or a disassortative property where degrees of connected vertices were anti-correlated.

Preferential Attachment: Graphs consist of 'hubs' of highly connected vertices with many small degree vertices connected to them (i.e. the rich get richer).

To remedy the inadequacies of the Erdős-Rényi random graph model in explaining real world graphs there have been many attempts to create ensembles that have some or all of these properties [9, 18, 23, 26, 29, 82–84, 98, 131]. Most [9, 26, 29, 82, 84, 131] break either the tree-like property or degree independence required for the analysis in subsequent chapters to work. However, there are some ensembles that are able to include some or all of these properties into typical draws from that ensemble whilst preserving the requirements for predictions using the cavity method. All of these can be viewed as extending the Erdős-Rényi ensemble in some way.

The most obvious way to extend the Erdős-Rényi ensemble so that we may generate different degree distributions is the generalised random graph ensemble [23]. In this ensemble the independent Bernoulli distribution is no longer identical for each potential edge, (i, j). We define the generalised random graph ensemble following Britton et al. [23] as follows:

Definition 2.4.3 (Generalised Random Graph Ensemble[23]:)**.** The generalised random graph ensemble, $\mathcal{G}_{V,\boldsymbol{w}}$, with $\boldsymbol{w} \in \mathbb{R}^V$, is defined as the set of all labelled graphs $\mathcal{G}^V$ endowed with probability distribution

$$P_{\mathcal{G}}(G \in \mathcal{G}^V) = \prod_{i<j}\left[\left(\frac{w_i w_j}{\sum_k w_k + w_i w_j}\right)a_{ij} + \left(1 - \frac{w_i w_j}{\sum_k w_k + w_i w_j}\right)(1 - a_{ij})\right] \quad (2.4.3)$$

One can see that for $\boldsymbol{w} = (\frac{Vp}{1-p}, \ldots, \frac{Vp}{1-p})^{\mathrm{T}}$ this distribution becomes the Erdős-Rényi ensemble, $\mathcal{G}_{V,p}$, once more. Similar definitions in Chung and Lu [26], Norros and Reittu [84] can be shown to be equal to definition 2.4.3 in the limit of large numbers of vertices [56].

Britton et al. [23] extended the generalised random graph ensemble further by considering the case where w_i were i.i.d. random variables with mean μ_w and finite $1 + \epsilon$ moment for some $\epsilon > 0$. In this case the authors proved the following facts [23, Theorem 3.1]

(a) The limiting distribution of a degree variable d_k as V becomes large is a mixed Poisson with parameter w_k.

(b) For any m, the degrees $d_1, \ldots, d_m$ are asymptotically independent.

In the main text we consider power law random graphs. These graphs are generalised random graphs with w_i sampled from a Pareto distribution. They are defined as follows

Definition 2.4.4 (Power law random graph ensemble:)**.** The power law random graph ensemble, $\mathcal{G}^{\text{pow}}_{V,\alpha,\lambda_c}$, with exponent $\alpha > 1$ and shift $\lambda_c > 0$ is defined as the set of all labelled graphs $\mathcal{G}^V$ endowed with probability distribution

$$P_{\mathcal{G}}(G \in \mathcal{G}^V) = \int \mathrm{d}\boldsymbol{w}\, p(\boldsymbol{w}) \prod_{i<j} \left[\left(\frac{w_i w_j}{\sum_k w_k + w_i w_j} \right) a_{ij} + \left(1 - \frac{w_i w_j}{\sum_k w_k + w_i w_j} \right) (1 - a_{ij}) \right] \tag{2.4.4}$$

where

$$p(\boldsymbol{w}) = \prod_i \frac{\alpha \lambda_c^\alpha}{w_i^{\alpha+1}} \theta(w_i - \lambda_c) \tag{2.4.5}$$

Using property (a) from Britton et al. [23, Theorem 3.1] we see that the resulting large V degree distribution for power law random graphs will become

$$p(d) = \int_{\lambda_c}^{\infty} \mathrm{d}\lambda\, \frac{\lambda^d \exp(-\lambda)}{d!} \frac{\alpha \lambda_c^\alpha}{\lambda^{\alpha+1}}. \tag{2.4.6}$$

A sample from the power law random graph ensemble is given in figure 2.9 along with the predicted degree distribution. Details on how to generate graphs from this ensemble are given in algorithm 2.2.

2.4.3 d-Regular Random Graphs

To be able to understand the basic properties of the random walk kernel without obstruction from the topology of a graph the final ensemble we will consider in subsequent chapters is the d-regular graph ensemble. This is the original model assumed in Viana and Bray [126].

In the d-regular graph ensemble each vertex degree is set fixed and equal to d. We define the d-random regular graph ensemble as follows

Algorithm 2.2: Power law random graph generator

Input : Number of vertices V, the exponent of the Pareto distribution α and the shift in the distribution α_s

Output: Graph $G(\mathcal{V}, \mathcal{E})$

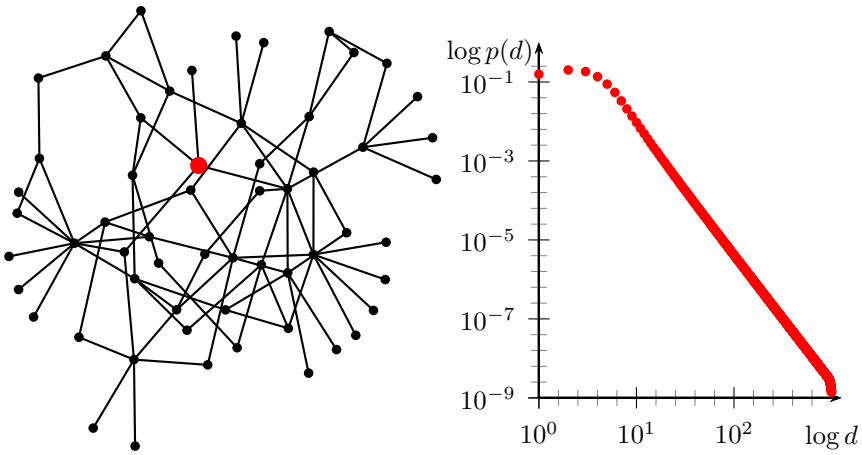
```
1  begin
2      Initialise vertex set V ← {1, ..., V};
3      Initialise edge set E ← ∅;
4      Initialise w to be V samples from a Pareto distribution (see algorithm 2.3);
5      for i ← 1 to V do
6          for j ← 1 to i − 1 do
7              x ← Uniform(0, 1);
8              if x < w_i w_j / (Σ_k w_k + w_i w_j) then
9                  Add edge (i, j) to E;
10             end
11         end
12     end
13 end
```

Algorithm 2.3: Sampling from the Pareto distribution [35, Chapter 2]

Input : Exponent α and shift α_s

Output: x, a sample from the distribution

```
1  begin
2      Sample t ← Uniform(0, 1);
3      x ← α_s/(1 − t)^{1/α};
4  end
```



(a) (b)

Figure 2.9: (a) An expansion around a vertex (shown in red) inside the 'giant component' of a power law random graph with exponent, α, equal to 2.1, cutoff, λ_c, equal to two and five hundred vertices. The tree-like property, although not as obvious as figure 2.10 and figure 2.8(a), is still visible. (b) The expected degree distribution of the power law random graph ensemble (see (2.4.6)).

Definition 2.4.5 (d-Regular Random Graph Ensemble [126])**.** The d-regular random graph ensemble or regular random graph ensemble, $\mathcal{G}_{V,\mathrm{reg},d}$, with $d \in \mathbb{N}$, is defined as the set of all labelled graphs $\mathcal{G}^V$ endowed with probability distribution

$$P_{\mathcal{G}}(G \in \mathcal{G}^V) \propto \prod_{i<j}\left[\frac{d}{V}a_{ij} + \left(1 - \frac{d}{V}\right)(1 - a_{ij})\right]\prod_i \delta_{d_i,d}. \tag{2.4.7}$$

Like the Erdős-Rényi and generalised random graph ensembles the properties of graphs drawn from this ensemble have been studied in great detail; a summary can be found in Bollobás [22]. To be able to generate regular graphs uniformly we use the method presented in Steger and Wormald [118], summarised in algorithm 2.4. Graphs are constructed by creating d 'edge stubs' for each vertex. These stubs are randomly paired until a graph is created or no more pairings can be made. In the latter case the algorithm is restarted. A sample from the regular graph ensemble is shown in figure 2.10.

2.4.4 Arbitrary degree distributions

The results that we will derive in this thesis will hold true for any graph ensemble where degrees of each vertex are i.i.d. However, to make things concise we will focus on the

Algorithm 2.4: Regular random graph generator[118]

Input : Degree d

Output: Graph $G(\mathcal{V}, \mathcal{E})$

begin
  Initialise vertex set $\mathcal{V} \leftarrow \{1, \ldots, V\}$;
  while $G(\mathcal{V}, \mathcal{E})$ *not regular* **do**
    Initialise edge set $\mathcal{E} \leftarrow \emptyset$;
    Initialise $\mathcal{E}' \leftarrow \{(i,j) | i = 1, \ldots, V, \quad j = 1, \ldots, V, \quad j < i\}$;
    Initialise $\boldsymbol{d} \leftarrow (d, \ldots, d)^{\mathrm{T}}$, $|\boldsymbol{d}| = V$;
    while $|\mathcal{E}'| \neq 0$ **do**
      Randomly select (i,j) from $\mathcal{E}'$ with probability proportional to $d_i d_j$;
      $\mathcal{E} \leftarrow \mathcal{E} \cup \{(i,j)\}$;
      $d_i \leftarrow d_i - 1$;
      $d_j \leftarrow d_j - 1$;
      $\mathcal{E}' \leftarrow \mathcal{E}' \setminus \{(i,j)\}$;
    end
  end
end

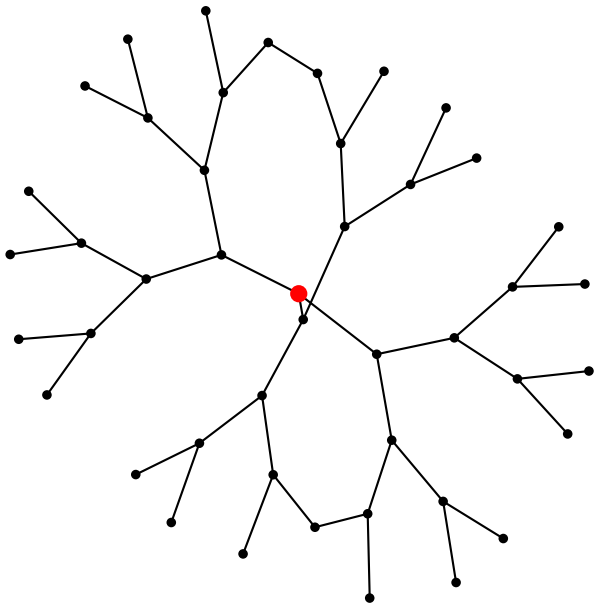


Figure 2.10: An expansion around a vertex (shown in red) in a regular random graph with five hundred vertices and degree equal to three. Tree-like structure is clearly visible

three ensembles mentioned above. For completeness here we will briefly discuss methods of generating random graph ensembles with an *arbitrary degree distribution*.

To create a graph from a degree distribution, $p(d)$, one first samples V degrees from $p(d)$; this is known as the degree sequence of the graph. Graphs are then constructed from the degree sequence.

Generating specific instance graphs for a degree sequence, $d_1, \ldots, d_V$, is relatively simple [127]. However, graphs created this way will not be generated uniformly from the ensemble of graphs with degree distribution $p(d)$. This is important for later results where the average over the ensemble is calculated numerically. One way to resolve this problem is to add a randomisation step after the graph has been created. This follows some form of Markovian edge switching dynamics. Originally this approach was noted by Eggleton and Holton [38], Taylor [121]. However, as pointed out in Coolen et al. [31], for degree distributions with $\langle d^2 \rangle d_{\max} / \langle d \rangle^2 \gg V$ this method still does not generate random graphs uniformly. Instead one requires a 'rejection' step [31].

CHAPTER 3

THE RANDOM WALK KERNEL AS A GAUSSIAN PROCESS PRIOR

We begin our discussion of GPs for regression with random walk kernels by first studying how the random walk kernel behaves as a covariance matrix of a GP prior. By studying the implications of a random walk kernel GP prior we will come across some non-trivial unexpected properties that will have implications in later cavity and replica calculations.

3.1 Insight into the random walk kernel

The random walk kernel derives its name from its interpretation in terms of walks on a graph. If we binomially expand the random walk kernel we have

$$\begin{aligned} \boldsymbol{C} = \left(\boldsymbol{I} - a^{-1}\boldsymbol{L}\right)^p &= \left((1-a^{-1})\boldsymbol{I} + a^{-1}\boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2}\right)^p \\ &= \sum_{q=0}^{p} \binom{p}{q} (1-a^{-1})^{p-q} (a^{-1})^q (\boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2})^q \\ &= \boldsymbol{D}^{-1/2} \sum_{q=0}^{p} \binom{p}{q} (1-a^{-1})^{p-q} (a^{-1})^q (\boldsymbol{A}\boldsymbol{D}^{-1})^q \boldsymbol{D}^{1/2}. \end{aligned} \tag{3.1.1}$$

where $\boldsymbol{D} = \mathrm{diag}(d_1, \ldots, d_V)^{\mathrm{T}}$ is a diagonal matrix of the degrees of the graph described by $\boldsymbol{A}$. In (3.1.1) the matrix $\boldsymbol{A}\boldsymbol{D}^{-1}$ is a random walk transition matrix. To see this assume there is a 'walker' at vertex i with degree d_i, then assuming the walker picks a neighbour to jump to with equal probability, the probability of the walker being at a neighbour j after taking a step is simply $1/d_i$. The probability of the walker being at any other vertex is zero. Writing this in terms of the adjacency matrix we have $a_{ij}/d_i = (\boldsymbol{A}\boldsymbol{D}^{-1})_{ij}$.

With this interpretation we see that apart from pre- and post-multiplication by $\boldsymbol{D}^{-1/2}$ and $\boldsymbol{D}^{1/2}$, the kernel $\boldsymbol{C}$ is a q-step random walk transition matrix, averaged over the number of steps q distributed according to $q \sim \mathrm{Binomial}(p, a^{-1})$. An alternative interpretation is to think of (3.1.1) in terms of a p-step lazy random walk, where at each step the walker stays at the current vertex with probability $(1 - a^{-1})$ and moves to a neighbouring vertex with probability $(da)^{-1}$.

Using either interpretation, one sees that p/a is the lengthscale over which the random walk can diffuse along the graph, and hence the lengthscale describing the typical maximum range of correlations of the random walk kernel (see figure 3.1). A peculiarity of this kernel is that as well as the maximum lengthscale p/a it has as suggested by the random walk kernel's relationship with the diffusion kernel for large p (see section 2.1.2) a diffusive lengthscale given by $(2p/a)^{1/2}$. This will appear in later chapters when we discuss the learning curves of GPs for regression with a random walk kernel.

For $\boldsymbol{C}$ to be a well behaved kernel we require that as the lengthscale diverges ($p \to \infty$ and $p/a \to \infty$) the kernel will approach a form for which knowing a single point in the graph tells us the entire function on the graph. This is known as the *fully correlated limit*. One can see that this is the case for the random walk kernel by observing that for large p, a random walk on a graph will approach a stationary distribution $\boldsymbol{p}_\infty = \boldsymbol{D}\boldsymbol{e}$, $\boldsymbol{e} = (1, \ldots, 1)^{\mathrm{T}}$ [see for example 70]. Since as p tends to infinity the mean of the binomial distribution tends to infinity and the variance tends to zero the q-step random walk is therefore $(\boldsymbol{A}\boldsymbol{D}^{-1})^q \approx \boldsymbol{p}_\infty \boldsymbol{e}^{\mathrm{T}} = \boldsymbol{D}\boldsymbol{e}\boldsymbol{e}^{\mathrm{T}}$, representing the fact that the random walk becomes stationary independently of the starting vertex [see for example 70]. This gives for p tending to infinity, the kernel $\boldsymbol{C} \propto \boldsymbol{D}^{1/2}\boldsymbol{e}\boldsymbol{e}^{\mathrm{T}}\boldsymbol{D}^{1/2}$, i.e. $C_{ij} \propto d_i^{1/2} d_j^{1/2}$, which corresponds to full correlation across vertices as expected; explicitly, if $\boldsymbol{f}$ is a GP on the

graph with covariance matrix $\boldsymbol{D}^{1/2}\boldsymbol{e}\boldsymbol{e}^{\mathrm{T}}\boldsymbol{D}^{1/2}$, then $\boldsymbol{f} = v\boldsymbol{D}^{1/2}\boldsymbol{e}$ with v a single Gaussian degree of freedom that could be determined from one noise free example.

3.1.1 Approaching the fully correlated kernel

Although calculating the fully correlated limit of the random walk kernel was straightforward we will now show that the convergence of the kernel to this limit is non-trivial. To understand the convergence of the kernel to the fully correlated limit we will study how the kernel behaves on a uniformly sampled d-regular graph (see section 2.4.3). We focus on graphs of this type because their regular structure will allow us to get a clear idea of the kernel behaviour without topological graph effects.

For a large enough number of vertices, V, typical simple cycles (a series of connected vertices starting and ending at the same point without passing through the same vertex twice) in a d-regular graph will be large, of length $O(\log V)$, and can be neglected for the calculation of the kernel as V tends to infinity. We therefore begin by assuming the graph is an infinite tree, and assess later how the cycles that do exist on random d-regular graphs cause departures from this picture.

An infinite vertex d-regular tree or *Bethe-lattice* [16] is a graph where each vertex has degree d with no cycles; it is unique up to permutations of vertices. Since all vertices on a tree are equivalent, the random walk kernel can only depend on the distance between vertices i and j, i.e. the smallest number of steps required to get from one vertex to the other. Denoting the value of a p-step unnormalised random walk kernel for vertices a distance l apart by $C_{l,p}$, we can determine these values by recursion over p as follows:

$$\begin{aligned} C_{l,p=0} &= \delta_{l,0}, \qquad \gamma_{p+1}C_{0,p+1} = \left(1 - \frac{1}{a}\right) C_{0,p} + \frac{1}{a}C_{1,p} \\ \gamma_{p+1}C_{l,p+1} &= \frac{1}{ad}C_{l-1,p} + \left(1 - \frac{1}{a}\right) C_{l,p} + \frac{d-1}{ad}C_{l+1,p} \quad l \geq 1. \end{aligned} \tag{3.1.2}$$

Here γ_p is chosen to achieve the desired normalisation for every p. To keep things concrete we will normalise so that $C_{0,p} = 1$.

Figure 3.1 shows the results obtained by iterating equation (3.1.2) numerically for a 3-regular tree with a equal to two. As expected the kernel becomes longer-ranged initially as p is increased, but seems to approach a non-trivial limiting form. This limiting form can be calculated using a modified version of the work presented in Chung and Yau [27]

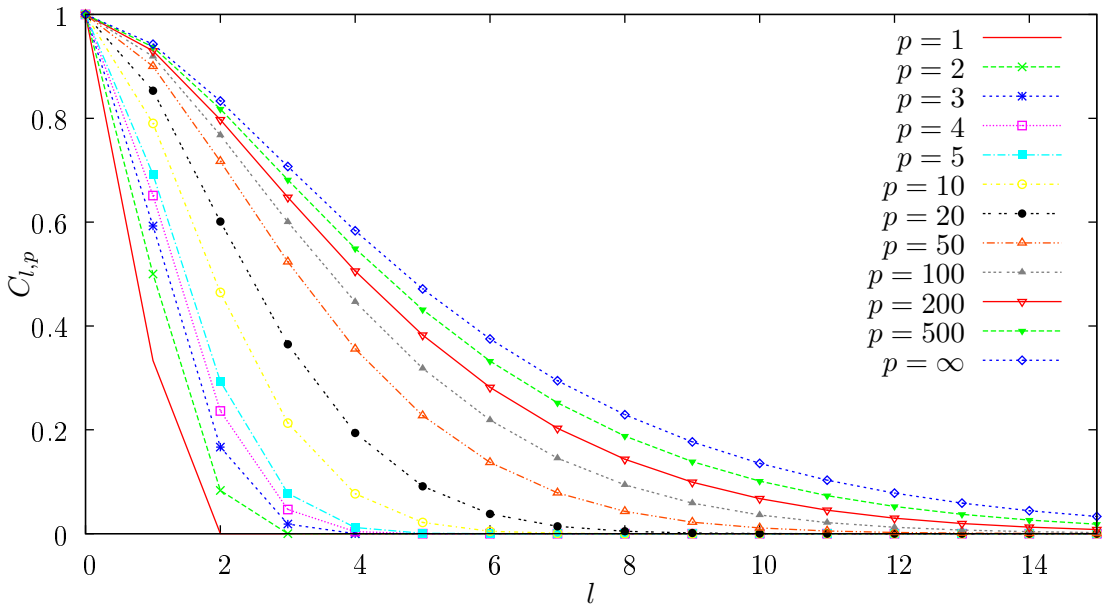


Figure 3.1: Random walk kernel $C_{l,p}$ on a 3-regular tree plotted against distance l for increasing number of steps p and $a = 2$.

(see Appendix A) and is given by

$$C_{l,p\to\infty} = \left[1 + \frac{l(d-2)}{d}\right] \frac{1}{(d-1)^{l/2}} \tag{3.1.3}$$

An alternative way of deriving this is to rewrite the random walk kernel in terms of shells, i.e. grouping vertices according to their distance l from a chosen central vertex. The number of vertices in the l-th shell, or shell volume, is $v_l = d(d-1)^{l-1}$ for $l \geq 1$ and $v_0 = 1$. Introducing $R_{l,p} = C_{l,p}\sqrt{v_l}$, equation (3.1.2) can be written in the form

$$\begin{aligned} R_{l,p=0} &= \delta_{l,0}, \qquad \gamma_{p+1} R_{0,p+1} = \left(1 - \frac{1}{a}\right) R_{0,p} + \frac{1}{a\sqrt{d}} R_{1,p} \\ \gamma_{p+1} R_{l,p+1} &= \frac{\sqrt{d-1}}{ad} R_{l-1,p} + \left(1 - \frac{1}{a}\right) R_{l,p} + \frac{\sqrt{d-1}}{ad} R_{l+1,p} \quad l \geq 1. \end{aligned} \tag{3.1.4}$$

Equation (3.1.4) is just the un-normalised diffusion equation for a biased random walk on a one dimensional lattice with a reflective boundary at zero. This has been solved in Monthus and Texier [79], and mapping this solution back to $C_{l,p}$ gives (3.1.3) (see section A.3 for further details).

This unexpected limiting form on a d-regular tree shows that, for large p, the random walk kernel does not approach the expected fully correlated limit in this case: because all vertices have the same degree this limit would correspond to $C_{l,p\to\infty}$ equal to unity.

On the other hand, on a d-regular graph with any finite number, V, of vertices the fully correlated limit must necessarily be approached as p tends to infinity[1]. As a large regular graph is only locally treelike, the difference must arise from the existence of long cycles in a regular graph.

To estimate when the existence of cycles will start to affect the kernel, consider first a d-regular tree truncated at depth l. This will have $V = 1+d\sum_{k=1}^{l-1}(d-1)^k = O(d(d-1)^{l-1})$ vertices. On a d-regular graph with the same number of vertices, we therefore expect to encounter cycles after the number of steps taken along the graph is of order l. In the random walk kernel the typical number of steps is p/a, so effects of cycles should appear once p/a becomes larger than

$$\frac{p}{a} \approx \frac{\log(V)}{\log(d-1)}. \tag{3.1.5}$$

Figure 3.2 shows a comparison between $C_{1,p}$ as calculated from equation (3.1.2) for a 3-regular tree and its analogue on random 3-regular graphs of finite size, which we call $K_{1,p}$. We define this analogue as the average of $C_{ij}/\sqrt{C_{ii}C_{jj}}$ over all pairs of neighbouring vertices on a fixed graph, averaged further over a number of randomly generated regular graphs. The square root accounts for the fact that local kernel values C_{ii} can vary slightly on a regular graph because of cycles, while they are the same for all vertices of a regular tree. Looking at figure 3.2 one sees that, as expected from the arguments above, the nearest neighbour kernel value for the 3-regular graph, $K_{1,p}$, coincides with its analogue $C_{1,p}$ on the 3-regular tree for small p. When p/a crosses the threshold (3.1.5), cycles in the regular graph become important and the two curves separate. For larger p, the kernel value for neighbouring vertices approaches the fully correlated limit $K_{1,p} \to 1$ on a regular graph, while on a regular tree one has the non-trivial limit $C_{1,p} \to 2\sqrt{d-1}/d$ from (3.1.3).

In conclusion, the analysis of the random walk kernel's approach to its fully correlated limit has shown that these kernels have an unusual dependence on their lengthscale p/a. In particular, kernel values for vertices a short distance apart can remain significantly below the fully correlated limit, even if p/a is large. That limit is approached only once p/a becomes larger than the graph size-dependent threshold (3.1.5), at which point cycles become important. We have focussed here on random regular graphs, but the same

[1] This is true because any finite graph will have a stationary distribution [70].

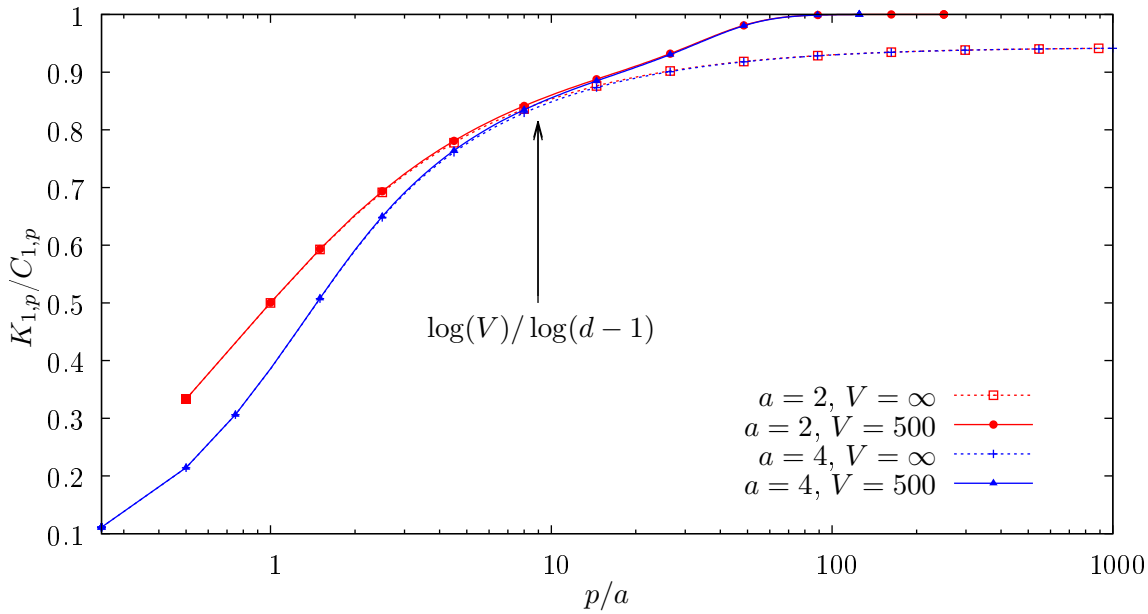


Figure 3.2: Comparison between numerical results for the average nearest neighbour kernel $K_{1,p}$ on random 3-regular graphs (solid line) with the result $C_{1,p}$ on a 3-regular tree (dashed line), calculated numerically by iteration of (3.1.1).

qualitative behaviour should be observed also on graphs with a non-trivial distribution of vertex degrees, $p(d)$.

3.2 Using the random walk kernel as a Gaussian process prior

Having gained a better understanding of the random walk kernel we now study its application in machine learning as a GP prior. We will focus in particular on how the kernel represents prior amplitude or scale of functions on the graph. For a GP this is given by the diagonal entries of the kernel.

Conventionally one fixes the desired scale of the kernel using *global normalisation*: denoting the unnormalised kernel by $\hat{C}(x, x')$ one scales $C(x, x') = \hat{C}(x, x')/\kappa$ with $\kappa = c^{-1}\langle \hat{C}(x, x)\rangle_x$ to achieve a desired average of $\langle C(x, x)\rangle_x = c$ across input locations x. In Euclidean spaces one typically uses translationally invariant kernels like the squared exponential kernel. For these spaces $C(x, x)$ is the same for all input locations

x and so global normalisation is sufficient to fix a spatially uniform scale for the prior amplitude. In the case of kernels on graphs on the other hand, the local connectivity structure around each vertex can be different. Since information about correlations 'propagates' only along graph edges, graph kernels are not generally translation invariant. In particular, there can be large variation among the prior variances at different vertices. This is usually undesirable in a probabilistic model unless one has strong prior knowledge to justify such variation. For the random walk kernel, the local prior variances are the diagonal entries of equation (3.1.1). These are directly related to the probability of return of a lazy random walk on a graph, which depends sensitively on the local graph structure. This dependence is in general non-trivial, and not just expressible through e.g. the degree of the local vertex. It seems difficult to imagine a scenario where such a link between prior variances and local graph structures could be justified by prior knowledge.

To emphasise the issue, figure 3.3(a) shows examples of distributions of local prior variances C_{ii} for random walk kernels globally normalised to an average prior variance of unity.[2] The distributions are peaked around the desired value of unity but contain many 'outliers' from vertices with abnormally low or high prior variance. Figure 3.3(a) shows the distribution of C_{ii} for a large single instance of an Erdős-Rényi random graph (see section 2.4.1). In such graphs, each edge is present independently of all others with some fixed probability, giving a Poisson distribution of degrees $p_\lambda(d) = \lambda^d \exp(-\lambda)/d!$; for the figure we chose an average degree, λ, equal to three. Figure 3.3(b) shows analogous results for a generalised random graph with power law mixing distribution (see section 2.4.2) with exponent, α, equal 2.5 and lower cutoff, λ_m, equal to two for the distribution of the means.

Looking first at figure 3.3(a), since we know that large Erdős-Rényi graphs are locally tree-like one might expect that this would lead to relatively uniform local prior variances. As shown in the figure, however, even for such tree-like graphs large variations can exist in the local prior variances. To give some specific examples, the large spike near zero is caused by single disconnected vertices and the smaller spike at around 6.8 arises from two-vertex (single edge) disconnected subgraphs. Single vertex subgraphs have an atyp-

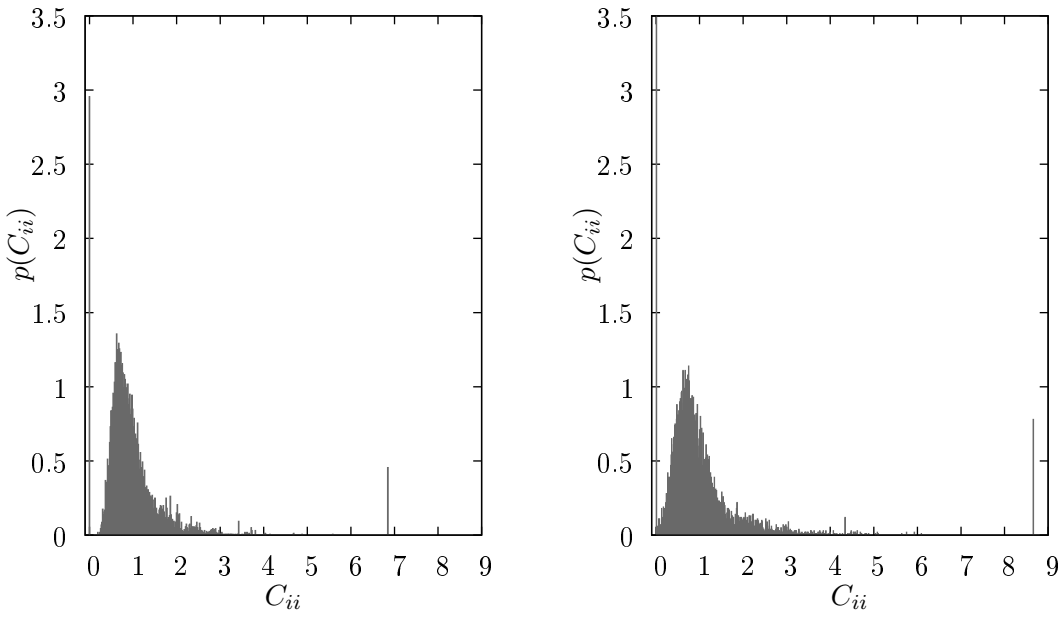
[2] We use C_{ii} again here, instead of $C(i,i)$ as in our general discussion of GPs; the subscript notation is more intuitive because the covariance function on a graph is just a $V \times V$ matrix.

ically small prior variance since for a single disconnected vertex i, before normalization $C_{ii} = (1 - a^{-1})^p$ which is the zero step ($q = 0$) contribution from equation (3.1.1) only. Other vertices in the graph will get additional contributions from q greater or equal to one and so have a larger prior variance. This effect will become more pronounced as p is increased and the binomial weights assign less weight to the q equal to zero term.

Somewhat surprisingly at first sight, the opposite effect is seen for two-vertex disconnected subgraphs as shown by the spike around $C_{ii} = 6.8$ in figure 3.3(a). For vertices on such subgraphs, $C_{ii} = \sum_{q=0}^{\lfloor p/2 \rfloor} \binom{p}{2q} a^{-2q}(1 - a^{-1})^{p-2q}$, which is an atypically large return probability: after any even number of steps, the walker must always return to its starting vertex. A similar situation would occur on vertices at the centre of a star. This illustrates that local properties of a vertex alone, like its degree, do not constrain the prior variance. In a two-vertex disconnected subgraph both vertices have degree one. But there will generically be other vertices of degree one that are dangling ends of a large connected graph component, and these will not have similarly elevated return probabilities. Thus, local graph structure is intertwined in a complex manner with local prior variance.

For the power law random graph, figure 3.3(a), the broad features of the distribution of local prior variances C_{ii} are similar: a peak at the desired value of unity, overlaid by spikes which again come from single and two-vertex disconnected subgraphs. The inset shows that decay from the mean is roughly exponential again, but with a slower decay; this is to be expected since power law graphs exhibit many more different local structures with a significantly larger probability than is the case for Erdős-Rényi graphs. Accordingly, the distribution of the C_{ii} also has a larger standard deviation than for the Erdős-Rényi case. The maximum values of C_{ii} that we see in these two specific graph instances follow the same trend, with the maximum prior variance approximately equal to forty for the power law graph and approximately equal to fifteen for the Erdős-Rényi graph. Such large values would constitute rather unrealistic prior assumptions about the scaling of the target function at these vertices.

To summarise, figure 3.3 shows that after global normalisation a random walk kernel can retain a large spread in the local prior variances, with the latter depending on the graph structure in a complicated manner. One approach to overcome this is to use a



(a) Erdős-Rényi (b) Power law

Figure 3.3: (a) Histogram of prior variances for the globally normalised random walk kernel with $a = 2$, $p = 10$ on a single instance of an Erdős-Rényi graph with mean degree $\lambda = 3$ and $V = 10000$ vertices. (b) As (a) but for a power law generalised random graph with exponent 2.5 and cutoff 2.

local normalisation. For a desired prior variance c this means normalising according to $C_{ij} = c\hat{C}_{ij}/(\kappa_i\kappa_j)^{1/2}$ with local normalisation constants $\kappa_i = \hat{C}_{ii}$; here $\hat{C}_{ij}$ is the unnormalized kernel matrix as before. This guarantees that all vertices have exactly equal prior variance as in the Euclidean case. No uncontrolled local variation in the scaling of the function prior then remains, and the computational overhead of local over global normalisation is negligible. Graphically, if we were to normalise the kernel to unity according to the local prescription, a plot of prior variances like the one in figure 3.3 would be a delta peak centred at 1.

It is worth noting here that this spread of prior variances would exist also if we had used the original definition of the normalised Laplacian seen in Chung [25], this definition differed from definition 2.1.4 in that it set $L_{ii} = 0$ for $d_i = 0$ (see footnote for definition 2.1.4); results for this case are shown in figure 3.4. In this case the disconnected single vertices have an unnormalised prior variance of one. Once the global normalisation is applied to this kernel the peak in the Erdős-Rényi graph in figure 3.3(a)

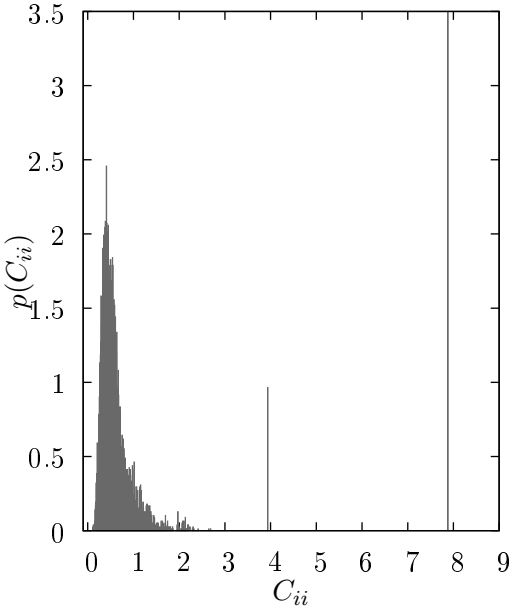


Figure 3.4: Histogram of prior variances for the globally normalised random walk kernel with the unaltered normalised Laplacian, $a = 2$ and $p = 10$ on a single instance of an Erdős-Rényi graph with mean degree $\lambda = 3$ and $V = 10000$ vertices.

is shifted to approximately $C_{ii} = 1/2$ and the large spike close to zero is shifted to the other extreme and is found at around $C_{ii} = 8$. For power law graphs these shifts are even more severe. Our argument in favour of local normalisation therefore can be applied to either definition of the normalised Laplacian.

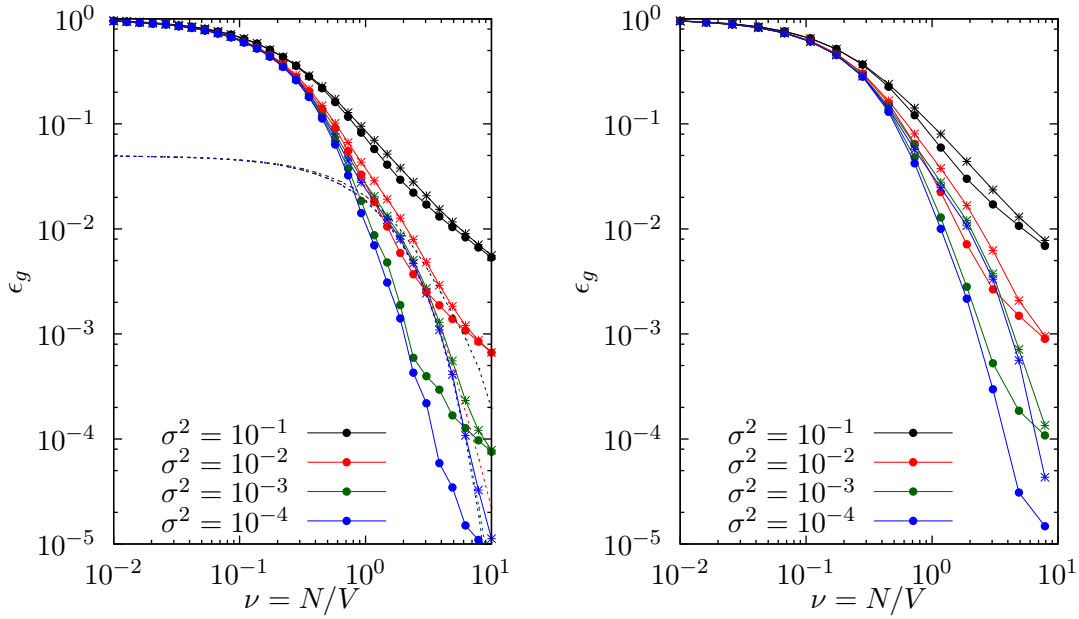
3.3 The effect of normalisation on learning

After arguing in favour of local normalisation we now qualitatively assess the implications of local over global kernel normalisation when using the random walk kernel as the covariance function of a GP prior. It is not a simple matter to say which kernel is 'better', the locally or globally normalised one. It would not make sense for instance to say the better kernel is the one that gives the lower Bayes error for given number of examples, as the Bayes error reflects both the complexity of the target function and the success in learning it. A more definite answer could be obtained only empirically, by running GP regression with local and global kernel normalization on the same datasets and comparing the prediction errors and also the marginal data likelihood. The same

approach could also be tried with synthetic datasets generated from GP priors that are mismatched to both priors we have considered, defined by the globally and locally normalised kernel, though justifying what is a reasonable choice for the prior of the target function would not be easy.

While a detailed study along the lines above is outside the scope of this thesis, we can nevertheless at least qualitatively study the effect of the kernel normalisation, to understand to what extent the corresponding priors define significantly different probabilistic models. Figure 3.5(a) and figure 3.5(b) overlay the learning curves for global and local kernel normalisations, for an Erdős-Rényi and a power law generalised random graph respectively. We will look at such learning curves in much more detail in the remainder of the thesis. For now the point is that there are qualitative differences in the shapes of the learning curves, with the ones for the locally normalised kernel exhibiting a shoulder around $\nu = 2$, where $\nu = N/V$ is the number of examples per vertex. This shoulder is due to the proper normalization of isolated vertices to unit prior variance; by contrast, as shown earlier in figure 3.3(a), global normalization gives too small a prior variance to such vertices. Figure 3.5(a) shows the expected learning curve contributions from all locally normalised isolated vertices (single vertex subgraphs) as dotted lines. After the GP learns the rest of the graph to a sufficient accuracy, the single vertex error dominates the learning curve until these vertices have typically seen at least one example. Once this point has been passed, the dominant error comes once more from the giant connected component of the graph, and the GP learns in a similar manner to the globally normalised case. A similar effect, although not plotted, is seen for the power law random graph case.

We can extend the scope of this qualitative comparison by examining how a student GP with a kernel with one normalisation performs when learning from a teacher with a kernel with the other normalisation. This is a case of *model mismatch* which we will address in more detail in Chapter 6 for the case of mismatched hyperparameters. Figure 3.6(a) shows the case of GP students with a globally normalised kernel for a teacher with a locally normalised kernel. The learning curves for the mismatched scenario are very different from those for the matched case (figure 3.5(a) and figure 3.5(b)), showing an increase in error as ν approaches unity. The resulting maximum in the learning curve



(a) Erdős-Rényi

(b) Power law

Figure 3.5: (a) Comparison between learning curves for locally (squares) and globally (circles) normalised kernels for Erdős-Rényi random graphs. (b) as (a) for power law random graphs.

again emphasises that the two choices of normalization produce distinctly different probabilistic models. Similar behaviour can be observed for the case of power law generalised random graphs, shown in figure 3.6(b). In both cases close inspection shows that the error maximum is caused by 'dangling edges' of the graph, i.e. chains of vertices (with degree two) extending away from the giant graph component and terminating in a vertex of degree one.

To understand this consider the mean of the predictive distribution i.e. the Bayes predictor after seeing N examples. If we arrange the predictive means of each vertex in a vector $\hat{\boldsymbol{f}} = (\hat{f}(1), \ldots, \hat{f}(V))^{\mathrm{T}}$, then from (2.1.10) we have

$$\hat{\boldsymbol{f}} = \boldsymbol{K}_V(\boldsymbol{K} + \sigma^2\boldsymbol{I})^{-1}\boldsymbol{y} \tag{3.3.1}$$

where $(\boldsymbol{K}_V)_{j,\mu} = C_{j,x_\mu}$, $\mu = 1, \ldots, N$ and $j = 1, \ldots, V$. We can rewrite this in the form $\hat{\boldsymbol{f}} = \boldsymbol{M}\boldsymbol{z}$ where

$$\boldsymbol{M} = \boldsymbol{K}_V(\boldsymbol{K}_f + \sigma^2\boldsymbol{I})^{-1/2} \tag{3.3.2}$$

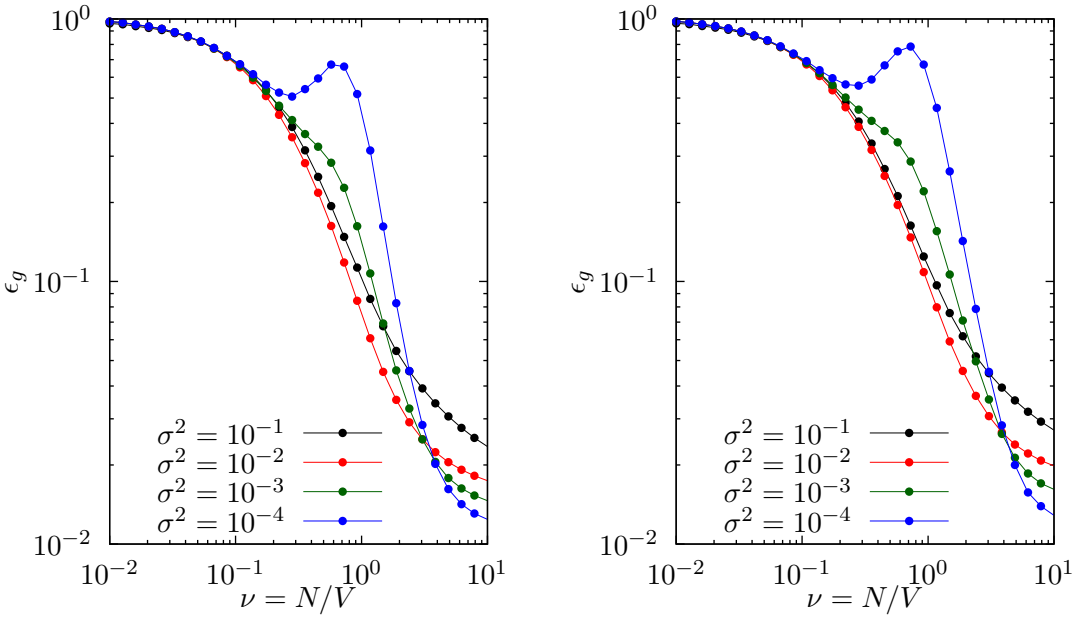
$$\boldsymbol{z} = (\boldsymbol{K}_f + \sigma^2\boldsymbol{I})^{-1/2}\boldsymbol{y}. \tag{3.3.3}$$

From this one sees that $\boldsymbol{M}$ represents the teacher-independent aspects of $\hat{\boldsymbol{f}}$ and the vector $\boldsymbol{z}$ of 'pseudo-training outputs' has been defined so that in the matched case it obeys $\langle \boldsymbol{z}\boldsymbol{z}^{\mathrm{T}} \rangle = \boldsymbol{I}$. We can think of $\boldsymbol{z}$ as pseudo training outputs because if its components z_μ are sampled i.i.d. from unit variance Gaussian variables, then $\hat{\boldsymbol{f}} = \boldsymbol{M}\boldsymbol{z}$ will have the correct distribution of mean prediction vectors. The columns $\boldsymbol{m}_1, \ldots, \boldsymbol{m}_N$ of $\boldsymbol{M}$ represent 'effective prediction vectors' conjugate to z_μ. In the matched case each $\hat{f}_i^2$ will then be, on average, the sum of the squares of the corresponding entries in the effective prediction vectors.

For the case of mismatch the only change is in the statistics of the $\boldsymbol{z}$. Their covariance matrix $\langle \boldsymbol{z}\boldsymbol{z}^{\mathrm{T}} \rangle$ will no longer be the identity matrix, and so act as a re-weighting of the prediction vectors. To understand our numerical simulations, we considered the values of ν around the maximum in the mismatched learning curve, listed the largest (diagonal) entries of $\boldsymbol{z}\boldsymbol{z}^{\mathrm{T}}$ and plotted their corresponding prediction vectors. These prediction vectors are generally localised around the dangling ends of the graph, thus substantiating that these are the cause of the bumps in figure 3.6.

As a final qualitative comparison between globally and locally normalised kernels, figure 3.7(a) and figure 3.7(b) show the variance of local posterior variances. This measures how much the local Bayes error typically varies from vertex-to-vertex, as a function of the data set size. Plausibly one would expect that this error variance is low initially when prediction on all vertices is equally uncertain. For large datasets the same should be true because errors on all vertices are then low. In an intermediate regime the error variance should become larger because examples will have been received on or near some vertices but not others. As figure 3.7(a) shows, for kernels with local normalization we find exactly this scenario, both for Erdős-Rényi and power law random graphs. The error variance is low for small $\nu = N/V$, increasing to a peak at $\nu \approx 0.2$ and finally decreasing again.

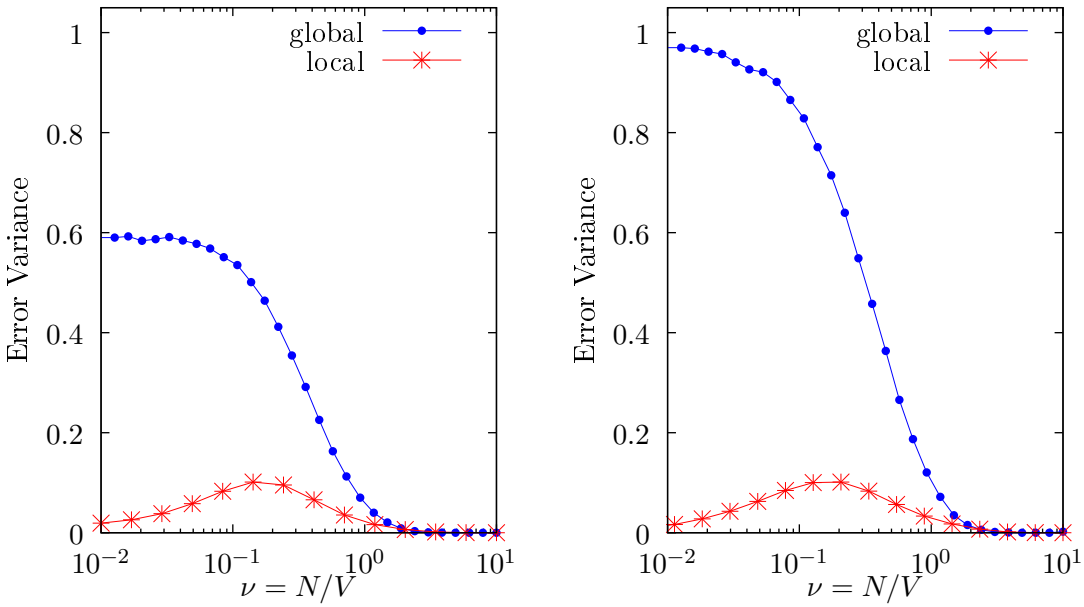
These results can now be contrasted with those for globally normalised kernels, also displayed in figure 3.7(a). Here the error variance is largest at ν equal to zero and decays from there. This means that the initial variance in the local prior variances is so large that any effects from the uneven distribution of example locations in any given data set remains subdominant throughout. We regard this as another indication of the probabilistically implausible character of the large spread of prior variances caused by



(a) Erdős-Rényi

(b) Power law

Figure 3.6: (a) Numerically simulated learning curves for a GP with a globally normalised kernel with $p = 10$ and $a = 2$, on Erdős-Rényi random graphs with mean degree 3 for a range of noise levels. The teacher GP has a locally normalised kernel with the same parameters. (b) as (a) but for power law generalised random graphs with exponent 2.5 and cutoff 2.



(a) Erdős-Rényi (b) Power law

Figure 3.7: (a) Error variance for GP regression with locally (stars) and globally (circles) normalised kernels against $\nu = N/V$, on Erdős-Rényi random graphs, and for matched learning with $p = 10$, $a = 2$, $\sigma^2 = 0.1$. (b) as (a) but for power law generalised random graphs.

global normalization.

In the subsequent chapters of this thesis we will try and approximate, or indeed predict exactly, the learning curves for GP regression. By doing so we will be able to accurately analyse the behaviour of GPs for regression with random walk kernels. In the next chapter we will begin by deriving an approximation for the learning curves for both globally and locally normalised random walk kernels in a matched scenario. We will show that local normalisation in particular leads to some complicated relationships between vertices that makes the learning curve difficult to calculate. To understand these complicated relationships we derive an equivalent replica method in Chapter 5. Finally, having gained a deeper understanding of the matched scenario, in Chapter 6 we will consider the case of mismatched kernels.

CHAPTER 4

APPROXIMATING THE LEARNING CURVE – THE CAVITY METHOD

The generalisation error, ϵ_g, as a function of the number of examples, N, also known as the *learning curve*, gives a wealth of knowledge about the performance of a learning method. One of its main uses is to give an indication of the number of examples required for the model to make accurate predictions for new data points. As was discussed in section 2.1.4.2, however, the learning curve is hard to calculate directly. In this chapter we will derive an approximation of the generalisation error for GP regression on graphs as a function of N using a statistical mechanics approach similar in flavour to that seen in Rogers et al. [97]. To do this we will require the use of the cavity method as discussed in section 2.3. We will begin with a brief recap of the main equations from Chapter 2 that we require to derive this approximation.

4.1 The graph specific generalisation error

We will focus in this chapter on a prior defined by a GP with covariance function given by the random walk kernel with parameters a and p. The random walk kernel is given

by

$$\hat{\boldsymbol{C}} = (\boldsymbol{I} - a^{-1}\boldsymbol{L})^p = (\boldsymbol{I}(1 - a^{-1}) + a^{-1}\boldsymbol{D}^{1/2}\boldsymbol{A}\boldsymbol{D}^{1/2})^p \tag{4.1.1}$$

where $\boldsymbol{L}$ is the normalised Laplacian (see (2.1.25)), $\boldsymbol{A}$ is the adjacency matrix of a graph (see definition 2.1.3) and $\boldsymbol{D} = \mathrm{diag}(d_1, \ldots, d_V)$ is a diagonal matrix of the degrees of the vertices of the graph described by $\boldsymbol{A}$. Since for a graph the kernel $\hat{\boldsymbol{C}}$ is just a $V \times V$ matrix we will refer to its entries using the convention $\hat{C}(i,j) = \hat{C}_{ij}$.

To be able to set the scale of the function the GP prior favours we will need to normalise the kernel. To encompass both the global and local normalisations discussed in Chapter 3 we wil consider a final GP covariance function (matrix) given by

$$\boldsymbol{C} = \boldsymbol{\mathcal{K}}^{-1/2}\hat{\boldsymbol{C}}\boldsymbol{\mathcal{K}}^{-1/2} \tag{4.1.2}$$

where $\boldsymbol{\mathcal{K}} = \mathrm{diag}(\kappa_1, \ldots, \kappa_V)$ is a normalising matrix. Under this formalism global normalisation corresponds to setting κ_i equal to κ, a constant proportional to the sum of the diagonal entries of $\hat{\boldsymbol{C}}$, i.e. $\kappa \propto \frac{1}{V}\sum_i \hat{\boldsymbol{C}}_{ii}$. Local normalisation then corresponds to setting κ_i to be proportional to the i-th entry of the diagonal elements of $\hat{\boldsymbol{C}}$, i.e. $\kappa_i \propto C_{ii}$. For both normalisations we will consider the case where the proportionality factor is set to one so that we predict functions with scale fixed to unity.

For a covariance function defined on a graph the GP prior is just a Gaussian distribution and will be given by

$$P(\boldsymbol{f}) \propto \exp\left(-\frac{1}{2}\boldsymbol{f}^{\mathrm{T}}\boldsymbol{C}^{-1}\boldsymbol{f}\right) \tag{4.1.3}$$

where we have again taken advantage of the fact that we are on a graph so that the function f is really a vector $\boldsymbol{f} = (f(1), \ldots, f(V))^{\mathrm{T}}$. Assuming we are presented inputs $\boldsymbol{X} = (x_1, \ldots, x_N)^{\mathrm{T}}$ with corresponding outputs $\boldsymbol{y} = (y_1, \ldots, y_N)^{\mathrm{T}}$ corrupted by Gaussian distributed noise with variance σ^2 we see, by applying Bayes theorem (see (2.1.2)), that the GP posterior distribution is again a Gaussian given by

$$P(\boldsymbol{f}|\boldsymbol{X}, \boldsymbol{y}) = \frac{\exp\left(-\frac{1}{2}\boldsymbol{f}^{\mathrm{T}}\boldsymbol{C}^{-1}\boldsymbol{f} - \frac{1}{2\sigma^2}\sum_{\mu=1}^{N}(f_{x_\mu} - y_\mu)^2\right)}{\int \mathrm{d}\boldsymbol{f}' \exp\left(-\frac{1}{2}(\boldsymbol{f}')^{\mathrm{T}}\boldsymbol{C}^{-1}\boldsymbol{f}' - \frac{1}{2\sigma^2}\sum_{\mu=1}^{N}(f'_{x_\mu} - y_\mu)^2\right)} \tag{4.1.4}$$

In this chapter we will assume that teacher and student are matched (see section 2.1.4.2).

Subject to this constraint the generalisation error for N examples is then [96]

$$\epsilon_g(N) = \left\langle \left\langle \frac{1}{V}\sum_{i=1}^{V}\int \mathrm{d}\boldsymbol{f}\,(\langle f_i\rangle_{\boldsymbol{f}|\boldsymbol{y},\boldsymbol{X}} - f_i)^2 \right.\right. \\ \left.\left. \times \frac{\exp\left(-\frac{1}{2}\boldsymbol{f}^{\mathrm{T}}\boldsymbol{C}^{-1}\boldsymbol{f} - \frac{1}{2\sigma^2}\sum_{\mu=1}^{N}(f_{x_\mu}-y_\mu)^2\right)}{\int \mathrm{d}\boldsymbol{f}'\exp\left(-\frac{1}{2}(\boldsymbol{f}')^{\mathrm{T}}\boldsymbol{C}^{-1}\boldsymbol{f}' - \frac{1}{2\sigma^2}\sum_{\mu=1}^{N}(f'_{x_\mu}-y_\mu)^2\right)} \right\rangle_{\boldsymbol{X}} \right\rangle_{\mathcal{G}} \tag{4.1.5}$$

which is also known as the *Bayes error*.

4.2 Predicting the learning curve

Equation (4.1.5) is simply the posterior variance of the student GP averaged over inputs $\boldsymbol{X}$ and graphs $\mathcal{G}$ (see section 2.1.4.2). Since we only need this posterior variance we shall shift $\boldsymbol{f}$ so that the posterior mean is $\boldsymbol{0}$; f_i is then just the deviation of the function value at vertex i from the posterior mean. The resulting Bayes error after shifting $\boldsymbol{f}$ becomes

$$\epsilon(N) = \left\langle \left\langle \frac{1}{V}\sum_{i=1}^{V}\int \mathrm{d}\boldsymbol{f}\, f_i^2 \frac{\exp\left(-\frac{1}{2}\boldsymbol{f}^{\mathrm{T}}\boldsymbol{C}^{-1}\boldsymbol{f} - \frac{1}{2\sigma^2}\sum_{\mu=1}^{N} f_{x_\mu}^2\right)}{\int \mathrm{d}\boldsymbol{f}\exp\left(-\frac{1}{2}\boldsymbol{f}^{\mathrm{T}}\boldsymbol{C}^{-1}\boldsymbol{f} - \frac{1}{2\sigma^2}\sum_{\mu=1}^{N} f_{x_\mu}^2\right)} \right\rangle_{\boldsymbol{X}} \right\rangle_{\mathcal{G}}. \tag{4.2.1}$$

which is just the posterior distribution with the linear terms dropped from the exponents. One should note that by shifting the posterior distribution to zero mean, we have eliminated the dependence on $\boldsymbol{y}$ in the above equation. That this should be so can also be seen from the covariance of the GP posterior (2.1.9), which only depends on training inputs x but not the corresponding outputs $\boldsymbol{y}$.

The averages in (4.2.1) are in general difficult to calculate analytically because the training input locations, x, enter in a highly nonlinear manner, see (2.1.9). We will now show, however, that in the case of GP regression of functions defined on graphs, generalisation errors can be predicted exactly in the limit of large random graphs using the cavity method. This prediction is broadly applicable and can be applied to all graph ensembles where $P(\mathcal{G})$ is characterised by a fixed degree distribution.

4.2.1 Creating the generating partition function

To begin our derivation of the approximation of the generalisation error we will first rewrite (4.2.1) in terms of a generating partition function, Z. Using the technique

discussed in section 2.2, we see that we may rewrite (4.2.1) in the form

$$\epsilon(N) = -\lim_{\lambda\to 0}\frac{2}{V}\frac{\partial}{\partial\lambda}\left\langle \log(Z)\right\rangle_{\boldsymbol{X},\mathcal{G}}, \tag{4.2.2}$$

with

$$Z = \int \mathrm{d}\boldsymbol{f}\,\exp\Big(-\frac{1}{2}\boldsymbol{f}^{\mathrm{T}}\boldsymbol{C}^{-1}\boldsymbol{f} - \frac{1}{2\sigma^2}\sum_{\mu=1}^{N} f_{x_\mu}^2 - \frac{\lambda}{2}\sum_i f_i^2\Big). \tag{4.2.3}$$

In order to apply the cavity method to the problem of predicting the generalisation error we must first rewrite the partition function (4.2.3) in the form of a graphical model. As discussed in section 2.3, this means that the weights being integrated over to obtain Z must consist of factors relating only to individual vertices, or to pairs of neighbouring vertices. The current form of (4.2.3) presents two problems to this analysis: the inverse of the covariance matrix in (4.2.3) creates factors linking vertices at arbitrary distances along the graph, and the likelihood term is not currently in a vertex specific form. To rewrite the likelihood term into a local vertex term we introduce γ_i, a count of the number of examples presented to vertex i. To eliminate interactions across the entire graph we Fourier transform the prior term $\exp(-\frac{1}{2}\boldsymbol{f}^{\mathrm{T}}\boldsymbol{C}^{-1}\boldsymbol{f})$ in (4.2.3), introducing Fourier variables $\boldsymbol{h}$, and then integrate out the remaining terms with respect to $\boldsymbol{f}$ to give

$$\begin{aligned} Z &\propto \int \mathrm{d}\boldsymbol{h}\mathrm{d}\boldsymbol{f}\,\exp\left(-\frac{1}{2}\boldsymbol{h}^{\mathrm{T}}\boldsymbol{C}\boldsymbol{h} + \mathrm{i}\boldsymbol{f}^{\mathrm{T}}\boldsymbol{h} - \frac{1}{2}\boldsymbol{f}^{\mathrm{T}}\mathrm{diag}\left(\left(\frac{\gamma_i}{\sigma^2}+\lambda\right)\right)\boldsymbol{f}\right) \\ &\propto \left[\prod_i\left(\frac{\gamma_i}{\sigma^2}+\lambda\right)^{-\frac{1}{2}}\right]\int \mathrm{d}\boldsymbol{h}\exp\left(-\frac{1}{2}\boldsymbol{h}^{T}\boldsymbol{C}\boldsymbol{h} - \frac{1}{2}\boldsymbol{h}^{T}\mathrm{diag}\left(\left(\frac{\gamma_i}{\sigma^2}+\lambda\right)^{-1}\right)\boldsymbol{h}\right). \end{aligned} \tag{4.2.4}$$

The coupling between different vertices in (4.2.4) is now through the covariance matrix, $\boldsymbol{C}$, and therefore still links vertices up to a distance p from each other. To reduce these remaining interactions to ones among nearest neighbours only, we must exploit the binomial expansion of the random walk kernel given in equation (3.1.1). Defining p additional variables at each vertex by $\boldsymbol{h}^q = \boldsymbol{\mathcal{K}}^{1/2}(\boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2})^q\boldsymbol{\mathcal{K}}^{-1/2}\boldsymbol{h}$, $q = 1,\ldots,p$, and abbreviating the coefficients by $c_q = \binom{p}{q}(1-a^{-1})^{p-q}(a^{-1})^q$, the interaction term $\boldsymbol{h}^T\boldsymbol{C}\boldsymbol{h}$ turns into a local term $\sum_{q=0}^{p} c_q(\boldsymbol{h}^0)^T\boldsymbol{\mathcal{K}}^{-1}\boldsymbol{h}^q$. (Here we have, for the sake of uniformity, written $\boldsymbol{h}^0$ instead of $\boldsymbol{h}$.) Of course the interactions have only been 'hidden' in the $\boldsymbol{h}^q$, but the key point is that the definition of these additional variables can be enforced recursively, via $\boldsymbol{h}^q = \boldsymbol{\mathcal{K}}^{1/2}\boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2}\boldsymbol{\mathcal{K}}^{-1/2}\boldsymbol{h}^{q-1}$. We represent this

definition via a Dirac delta function (for each $q = 1, \ldots, p$) and then Fourier transform the latter, with conjugate variables $\hat{\boldsymbol{h}}^q$, to get

$$Z \propto \prod_i \left(\frac{\gamma_i}{\sigma^2} + \lambda\right)^{-\frac{1}{2}} \int \prod_{q=0}^{p} \mathrm{d}\boldsymbol{h}^q \prod_{q=1}^{p} \mathrm{d}\hat{\boldsymbol{h}}^q \exp\left(-\frac{1}{2}(\boldsymbol{h}^0)^{\mathrm{T}} \mathrm{diag}\left(\left(\frac{\gamma_i}{\sigma^2} + \lambda\right)^{-1}\right) \boldsymbol{h}^0 \right.$$
$$\left. - \frac{1}{2}\sum_{q=0}^{p} c_q (\boldsymbol{h}^0)^{\mathrm{T}} \boldsymbol{\mathcal{K}}^{-1} \boldsymbol{h}^q + \mathrm{i}(\hat{\boldsymbol{h}}^q)^{\mathrm{T}} \left(\boldsymbol{h}^q - \boldsymbol{\mathcal{K}}^{1/2} \boldsymbol{D}^{-1/2} \boldsymbol{A} \boldsymbol{D}^{-1/2} \boldsymbol{\mathcal{K}}^{-1/2} \boldsymbol{h}^{q-1}\right) \right). \quad (4.2.5)$$

Because the graph adjacency matrix $\boldsymbol{A}$ now appears at most linearly in the exponent, all interactions are between nearest neighbours only. We have thus expressed our partition function Z as one of a (complex-valued) graphical model at a cost of introducing $2p$ additional variables at each vertex.

4.2.2 Global normalisation

We can now apply belief propagation to the calculation of marginals for the above graphical model. We focus first on the simpler case of a globally normalised kernel where $\kappa_i = \kappa$ for all i. Rescaling each h_i^q to $d_i^{1/2}\kappa^{1/2} h_i^q$ and $\hat{h}_i^q$ to $d_i^{1/2}\hat{h}_i^q/\kappa^{1/2}$ we are left with

$$Z \propto \prod_i \left(\frac{\gamma_i}{\sigma^2} + \lambda\right)^{-\frac{1}{2}} \int \prod_{q=0}^{p} \mathrm{d}\boldsymbol{h}^q \prod_{q=1}^{p} \mathrm{d}\hat{\boldsymbol{h}}^q \prod_{(ij)} \exp\left(-\mathrm{i}\sum_{q=1}^{p} \left(\hat{h}_i^q h_j^{q-1} + \hat{h}_j^q h_i^{q-1}\right)\right)$$
$$\times \prod_i \exp\left(-\frac{1}{2}\sum_{q=0}^{p} c_q h_i^0 h_i^q d_i - \frac{1}{2}\frac{(h_i^0)^2 \kappa d_i}{\frac{\gamma_i}{\sigma^2} + \lambda} + \mathrm{i}\sum_{q=1}^{p} d_i \hat{h}_i^q h_i^q \right), \quad (4.2.6)$$

where the interaction terms coming from the adjacency matrix, $\boldsymbol{A}$, have been written explicitly as a product over distinct graph edges (i, j).

To see how the Bayes error (4.2.1) can be obtained from this partition function, we can differentiate $\log(Z)$ with respect to λ as prescribed by (4.2.2) to get

$$\epsilon(\nu) = \lim_{\lambda \to 0} \frac{1}{V} \sum_i \frac{1}{\frac{\gamma_i}{\sigma^2} + \lambda} \left(1 - \frac{d_i \kappa \langle (h_i^0)^2 \rangle}{\frac{\gamma_i}{\sigma^2} + \lambda}\right). \quad (4.2.7)$$

Equation (4.2.7) tells us that in order to calculate the Bayes error we require the marginal distributions of h_i^0. These can be calculated using the cavity method (see section 2.3): for a large random graph with arbitrary fixed degree sequence the graph is locally tree-like, so that if vertex i were eliminated the corresponding subgraphs (locally trees)

rooted at the neighbours $j \in \mathcal{N}(i)$ of i would become approximately independent. The resulting cavity marginals created by removing i, which we denote $P_j^{(i)}(\boldsymbol{h}_j, \hat{\boldsymbol{h}}_j|\boldsymbol{X})$ with $\boldsymbol{h}_j = (h_j^0, \ldots, h_j^p, \hat{h}_j^1, \ldots, \hat{h}_j^p)^{\mathrm{T}}$, can then be calculated iteratively within these subgraphs using the update equations

$$P_j^{(i)}(\boldsymbol{h}_j, \hat{\boldsymbol{h}}_j|\boldsymbol{X}) \propto \exp\left(-\frac{1}{2}\sum_{q=0}^{p} c_q d_j h_j^0 h_j^q - \frac{1}{2}\frac{d_j \kappa (h_j^0)^2}{\gamma_j/\sigma^2 + \lambda} + \mathrm{i}\sum_{q=1}^{p} d_j \hat{h}_j^q h_j^q\right)$$
$$\int \prod_{k\in\mathcal{N}(j)\setminus i} \mathrm{d}\boldsymbol{h}_k \mathrm{d}\hat{\boldsymbol{h}}_k \exp\left(-\mathrm{i}\sum_{q=1}^{p}(\hat{h}_j^q h_k^{q-1} + \hat{h}_k^q h_j^{q-1})\right) P_k^{(j)}(\boldsymbol{h}_k, \hat{\boldsymbol{h}}_k|\boldsymbol{X}). \quad (4.2.8)$$

In terms of the sum-product formulation of belief propagation, the cavity marginal on the *left hand side* (LHS) is the message that vertex j sends to the factor in Z for edge (i, j) [20] (see section 2.3 and figure 2.7).

One sees that the cavity update equations (4.2.8) are solved self-consistently by complex-valued Gaussian distributions with mean zero and covariance matrices $\boldsymbol{V}_j^{(i)}$. This Gaussian character of the solution was of course to be expected because in (4.2.6) we have a Gaussian graphical model. By performing the Gaussian integrals in the cavity update equations explicitly, one finds for the corresponding updates of the covariance matrices the rather simple form

$$\boldsymbol{V}_j^{(i)} = (\boldsymbol{O}_j - \sum_{k\in\mathcal{N}(j)\setminus i} \boldsymbol{X}\boldsymbol{V}_k^{(j)}\boldsymbol{X})^{-1}, \quad (4.2.9)$$

where we have defined the $(2p+1)\times(2p+1)$ matrices

$$\boldsymbol{O}_j = d_i \left(\begin{array}{cccc|ccc} c_0 + \frac{\kappa}{\gamma_i/\sigma^2+\lambda} & \frac{c_1}{2} & \cdots & \frac{c_p}{2} & 0 & \cdots & 0 \\ \frac{c_1}{2} & & & & -\mathrm{i} & & \\ \vdots & & & & & \ddots & \\ \frac{c_p}{2} & & & & & & -\mathrm{i} \\ \hline 0 & -\mathrm{i} & & & & & \\ \vdots & & \ddots & & & \mathbf{0}_{p,p} & \\ 0 & & & -\mathrm{i} & & & \end{array}\right) \quad (4.2.10)$$

$$\boldsymbol{X} = \left(\begin{array}{ccc|ccc} & & & \mathrm{i} & & \\ & \mathbf{0}_{p+1,p+1} & & & \ddots & \\ & & & & & \mathrm{i} \\ & & & 0 & \dots & 0 \\ \hline \mathrm{i} & & 0 & & & \\ & \ddots & \vdots & & \mathbf{0}_{p,p} & \\ & \mathrm{i} & 0 & & & \end{array}\right). \tag{4.2.11}$$

At first glance (4.2.9) seems to become singular as $\lambda \to 0$ for γ equal to zero; however this is easily avoided. We introduce $\boldsymbol{O}_j - \sum_{k=1}^{d-1} \boldsymbol{X}\boldsymbol{V}_k^{(j)}\boldsymbol{X} = \boldsymbol{M}_j + [d_j\kappa/(\gamma_j/\sigma^2+\lambda)]\boldsymbol{e}_0\boldsymbol{e}_0^{\mathrm{T}}$ with $\boldsymbol{e}_0^{\mathrm{T}} = (1,0,\dots,0)$ so that $\boldsymbol{M}_j$ contains all the non-singular terms. We may then apply the Woodbury identity [48] to write the matrix inverse in a form where the λ tending to zero limit can be taken without difficulties:

$$\left(\boldsymbol{O}_j - \sum_{k=1}^{d-1} \boldsymbol{X}\boldsymbol{V}_k^{(j)}\boldsymbol{X}\right)^{-1} = \boldsymbol{M}_j^{-1} - \frac{\boldsymbol{M}_j^{-1}\boldsymbol{e}_0\boldsymbol{e}_0^{\mathrm{T}}\boldsymbol{M}_j^{-1}}{(\gamma_j/\sigma^2+\lambda)/(d_j\kappa) + \boldsymbol{e}_0^{\mathrm{T}}\boldsymbol{M}_j^{-1}\boldsymbol{e}_0}. \tag{4.2.12}$$

In our derivation so far we have assumed a fixed graph. We must now translate these equations into the setting we ultimately want to study, i.e. an ensemble of large random graphs. We consider ensembles characterised by the distribution $p(d)$ of the degrees d_i, with every graph that has the desired degree distribution being assigned equal probability (this encompasses all the graph ensembles described in section 2.4). Instead of individual cavity covariance matrices $\boldsymbol{V}_j^{(i)}$, one must then consider their probability distribution $\pi(\boldsymbol{V})$ across all edges of the graph. Picking at random an edge (i,j) of a graph, the probability that vertex j will have degree d_j is then $p(d_j)d_j/\bar{d}$, because such a vertex has d_j 'chances' of being picked. (The normalisation factor is the average degree $\bar{d} = \sum_i p(d_i)d_i$.) Using again the locally treelike structure, the incoming (to vertex j) cavity covariances $\boldsymbol{V}_k^{(j)}$ will be i.i.d. samples from $\pi(\boldsymbol{V})$. Thus a fixed point of the cavity update equations corresponds to a fixed point of an update equation for $\pi(\boldsymbol{V})$:

$$\pi(\boldsymbol{V}) = \sum_d \frac{p(d)d}{\bar{d}} \left\langle \int \prod_{k=1}^{d-1} \mathrm{d}\boldsymbol{V}_k\, \pi(\boldsymbol{V}_k)\; \delta\left(\boldsymbol{V} - \left(\boldsymbol{O} - \sum_{k=1}^{d-1} \boldsymbol{X}\boldsymbol{V}_k\boldsymbol{X}\right)^{-1}\right)\right\rangle_{\gamma}. \tag{4.2.13}$$

Since the vertex label is now arbitrary, we have omitted the index j. The average in (4.2.13) is over the distribution of the number of examples $\gamma \equiv \gamma_j$ at vertex j. We will, as we did in Chapter 3 and will do throughout this thesis, assume for simplicity that

examples are drawn with uniform input probability across all vertices. This will mean that the distribution of γ is simply $\gamma \sim \text{Poisson}(\nu)$ in the limit of large N and V at fixed $\nu = N/V$.

In general equation (4.2.13) cannot be solved analytically, but we can tackle it numerically using population dynamics [78]. Population dynamics is an iterative technique where one creates a population of covariance matrices and for each iteration updates a random element of the population according to the delta function in (4.2.13). The update is calculated by sampling from the degree distribution $p(d)$ of local degrees, the Poisson distribution of the local number of examples ν and from the distribution $\pi(\boldsymbol{V}_k)$ of 'incoming' covariance matrices, the latter being approximated by uniform sampling from the current population. The method is summarised in algorithm 4.1.

Algorithm 4.1: Population dynamics [78]

Input : Population size M, number of iteration loops L, the expected number of examples per vertex ν and noise variance σ^2

Output: A population converged to the distribution $\pi(\boldsymbol{V})$

begin
  Initialise an array *pop* of size M of symmetric $2p+1 \times 2p+1$ random matrices;
  for $i = 0$; $i < ML$; i++ **do**
    Sample a degree d from $dp(d)/\bar{d}$;
    Sample the number of examples γ from a Poisson distribution with mean ν;
    Select d matrices $\boldsymbol{V}_1, \ldots, \boldsymbol{V}_d$ uniformly from *pop*;
    Set $\boldsymbol{V}_d \leftarrow \left(\boldsymbol{O} - \sum_{k=1}^{d-1} \boldsymbol{X}\boldsymbol{V}_k\boldsymbol{X}\right)^{-1}$;
  end
end

Once we have $\pi(\boldsymbol{V})$, the Bayes error can be found from the graph ensemble version of equation (4.2.2). This is obtained by inserting the explicit expression for $\langle (h_i^0)^2 \rangle$ in terms of the cavity marginals of the neighbouring vertices, and replacing the average

over vertices with an average over degrees d:

$$\epsilon(\nu) = \lim_{\lambda\to 0} \sum_d p(d) \Bigg\langle \frac{1}{\gamma/\sigma^2 + \lambda} \times \left(1 - \frac{d\kappa}{\gamma/\sigma^2+\lambda} \int \prod_{k=1}^{d} \mathrm{d}\boldsymbol{V}_k \, \pi(\boldsymbol{V}_k) \, (\boldsymbol{O} - \sum_{k=1}^{d} \boldsymbol{X}\boldsymbol{V}_k\boldsymbol{X})^{-1}_{00}\right)\Bigg\rangle_\gamma . \quad (4.2.14)$$

The number of examples at the vertex γ is once more to be averaged over $\gamma \sim \text{Poisson}(\nu)$. The subscript '00' indicates the top left element of the matrix, which determines the variance of h^0.

To be able to use equation (4.2.14), in a similar manner to (4.2.9), we will again need to rewrite into a form that remains explicitly non-singular when γ and λ equal zero. We separate the γ-dependence of the matrix inverse again and write, in slightly modified notation as appropriate for the graph ensemble case, $\boldsymbol{O} - \sum_{k=1}^{d} \boldsymbol{X}\boldsymbol{V}_k\boldsymbol{X} = \boldsymbol{M}_d + [d\kappa/(\gamma/\sigma^2+\lambda)]\boldsymbol{e}_0\boldsymbol{e}_0^{\mathrm{T}}$, where $\boldsymbol{e}_0^{\mathrm{T}} = (1, 0, \ldots, 0)$. The 00 element of the matrix inverse appearing above can then be expressed using the Woodbury formula [48] as

$$\begin{aligned} \boldsymbol{e}_0^{\mathrm{T}} \left(\boldsymbol{O} - \sum_{k=1}^{d} \boldsymbol{X}\boldsymbol{V}_k\boldsymbol{X}\right)^{-1} \boldsymbol{e}_0 &= \boldsymbol{e}_0\boldsymbol{M}_d^{-1}\boldsymbol{e}_0^{\mathrm{T}} - \frac{\boldsymbol{e}_0\boldsymbol{M}_d^{-1}\boldsymbol{e}_0\boldsymbol{e}_0^{\mathrm{T}}\boldsymbol{M}_d^{-1}\boldsymbol{e}_0^{\mathrm{T}}}{(\gamma/\sigma^2+\lambda)/(d\kappa) + \boldsymbol{e}_0^{\mathrm{T}}\boldsymbol{M}_d^{-1}\boldsymbol{e}_0} \\ &= \frac{\boldsymbol{e}_0^{\mathrm{T}}\boldsymbol{M}_d^{-1}\boldsymbol{e}_0(\gamma/\sigma^2+\lambda)/(d\kappa)}{(\gamma/\sigma^2+\lambda)/(d\kappa) + \boldsymbol{e}_0^{\mathrm{T}}\boldsymbol{M}_d^{-1}\boldsymbol{e}_0}. \end{aligned} \quad (4.2.15)$$

Rewritten in the form (4.2.15) the $\lambda \to 0$ limit can now be taken, with the result

$$\epsilon(\nu) = \left\langle \sum_d p(d) \int \prod_{k=1}^{d} \mathrm{d}\boldsymbol{V}_k \, \pi(\boldsymbol{V}_k) \, \frac{1}{\gamma/\sigma^2 + d\kappa(\boldsymbol{M}_d^{-1})_{00}} \right\rangle_\gamma . \quad (4.2.16)$$

This has a simple interpretation: the cavity marginals of the neighbours provide an effective Gaussian prior for each vertex, whose inverse variance is $d\kappa(\boldsymbol{M}^{-1})_{00}$. We can evaluate the integral in (4.2.16) using Monte-Carlo techniques by sampling from the population used in the update step [58]. The procedure for calculating (4.2.16) is summarised in algorithm 4.2.

The self-consistency equation (4.2.13) for $\pi(\boldsymbol{V})$ and the expression (4.2.16) for the resulting Bayes error allow us to predict learning curves as a function of the number of examples per vertex, ν, for *arbitrary degree distributions* $p(d)$ of our random graph ensemble. For large graphs these predictions should become exact. It is worth stressing that such exact learning curve predictions have previously only been available in very

Algorithm 4.2: Calculating equation (4.2.16)

Input : Converged population *pop*, number of iteration loops L, the expected number of examples per vertex ν and noise variance σ^2

Output: An approximation of $\epsilon_g(\nu)$ given in (4.2.16)

begin
  Initialise a running total $tot = 0$;
  Initialise a $2p+1 \times 2p+1$ matrix $\boldsymbol{M}_d^{-1}$;
  for $i = 0$; $i < L$; i++ **do**
    Sample a degree d from $p(d)$;
    Sample the number of examples γ from a Poisson distribution with mean ν;
    Select d matrices $\boldsymbol{V}_1, \ldots, \boldsymbol{V}_d$ uniformly from *pop*;
    Set $\boldsymbol{M}_d^{-1} \leftarrow \left(\boldsymbol{O} - \sum_{k=1}^{d} \boldsymbol{X}\boldsymbol{V}_k\boldsymbol{X} - [d\kappa/(\gamma/\sigma^2 + \lambda)]\boldsymbol{e}_0\boldsymbol{e}_0^{\mathrm{T}}\right)^{-1}$;
    Set $tot \leftarrow tot + 1/(\gamma/\sigma^2 + d\kappa(\boldsymbol{M}_d^{-1})_{00})$;
  end
  Return $\epsilon_g \leftarrow tot/L$;
end

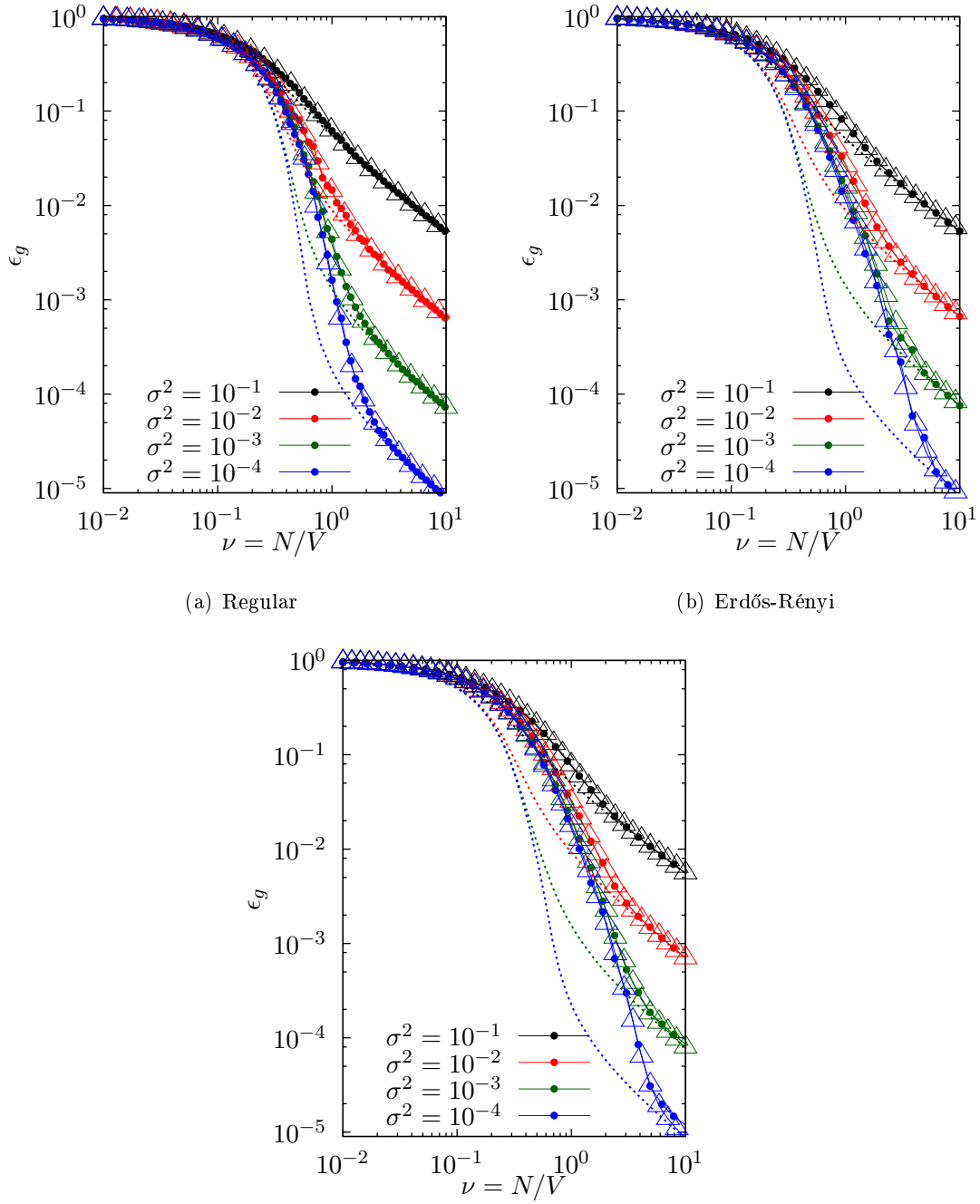
specific, noise-free, GP regression scenarios [72, 96], while our result for GP regression on graphs is applicable to a broad range of random graph ensembles, with arbitrary noise levels and kernel parameters.

We note briefly that for graphs with isolated vertices ($d = 0$), one has to be slightly careful: already in the definition of the covariance function (4.1.1) one should replace $\boldsymbol{D} \to \boldsymbol{D} + \delta \boldsymbol{I}$ to avoid division by zero, taking δ to zero at the end. For d equal to zero one then finds in the expression (4.2.16) that $(\boldsymbol{M}^{-1})_{00} = 1/(c_0\delta)$, where c_0 is defined before (4.2.5). As a consequence, $\kappa(\delta + d)(\boldsymbol{M}^{-1})_{00} = \kappa\delta(\boldsymbol{M}^{-1})_{00} = \kappa/c_0$. This is to be expected since isolated vertices each have a separate Gaussian prior with variance c_0/κ.

Equations (4.2.13) and (4.2.16) still require the normalisation constant, κ. The simplest way to calculate this is to run the population dynamics once for κ equal to one and ν equal to zero, i.e. an unnormalised kernel and no training data. The result for ϵ then just gives the average (over vertices) prior variance. With κ set to this value, one can then run the population dynamics for any ν to obtain the Bayes error prediction for GP regression with a globally normalised kernel.

Comparisons between the cavity prediction of the generalisation error, numerically simulated generalisation errors and the results of the baseline eigenvalue approximation (see section 2.2.1) as a function of ν are shown in figure 4.1, for regular, Erdős-Rényi and generalised random graphs with power law degree distributions respectively. As can be seen the cavity predictions greatly outperform the baseline eigenvalue approximation derived in section 2.2.1 and are accurate along the whole length of the curve. This confirms our expectation that the cavity approach will become exact on large graphs, although it is remarkable that the agreement is quantitatively so good already for graphs with only five hundred vertices.

To get a better understanding of the generalisation error as a function of ν we can look at the distribution of cavity covariances $\pi(\boldsymbol{V})$ for different ν. Figure 4.2 (a) and (b) show the log-distribution of V_{00} for random regular graphs (where each vertex has fixed equal degree) with degree three. For ν equal to 0.0001 the distribution appears to consist of delta peaks. This can be confirmed analytically by a Taylor expansion of (4.2.13) about ν equal to zero.



(a) Regular

(b) Erdős-Rényi

(c) Power law

Figure 4.1: (a) Learning curves for GP regression with globally normalised kernels with $p = 10$, $a = 2$ on 3-regular random graphs for a range of noise levels σ^2. Dashed lines: eigenvalue predictions (see section 2.2.1), solid lines with filled circles: numerically simulated learning curves for graphs of size $V = 500$, solid lines with triangles: cavity predictions. (b) As (a) for Erdős-Rényi random graphs with mean degree equal to three. (c) As (a) for power law generalised random graphs with exponent 2.5 and cutoff 2.

If we set $\pi(\boldsymbol{V}) = \pi_0(\boldsymbol{V}) + \nu\pi_1(\boldsymbol{V}) + O(\nu^2)$ and substitute this into (4.2.13) we have

$$\pi_0(\boldsymbol{V}) + \nu\pi_1(\boldsymbol{V}) + O(\nu^2) = \left\langle \int \prod_{k=1}^{d-1} \mathrm{d}\boldsymbol{V}_k \prod_{k=1}^{d-1} \left[\pi_0(\boldsymbol{V}_k) + \nu\pi_1(\boldsymbol{V}_k) + O(\nu^2)\right] \times \delta\left(\boldsymbol{V} - \left(\boldsymbol{O} - \sum_{k=1}^{d-1} \boldsymbol{X}\boldsymbol{V}_k\boldsymbol{X}\right)^{-1}\right)\right\rangle_{\gamma} \quad (4.2.17)$$

where the average over the degree has been dropped because we are considering a d-regular graph. We see that by rewriting the Poisson distribution over examples, γ, as an expansion up to $O(\nu^2)$, and equating terms, we find the following self consistent equations for π_0 and π_1

$$\pi_0(\boldsymbol{V}) = \int \prod_{k=1}^{d-1} \mathrm{d}\boldsymbol{V}_k \prod_{k=1}^{d-1} \pi_0(\boldsymbol{V}_k)\delta(\ldots)_{\gamma=0} \quad (4.2.18)$$

$$\pi_1(\boldsymbol{V}) = (d-1)\int \prod_{k=1}^{d-1} \mathrm{d}\boldsymbol{V}_k \prod_{k=1}^{d-2} \pi_0(\boldsymbol{V}_k)\pi_1(\boldsymbol{V}_{d-1})\delta(\ldots)_{\gamma=0} - \pi_0(\boldsymbol{V}) + \int \prod_{k=1}^{d-1} \mathrm{d}\boldsymbol{V}_k \prod_{k=1}^{d-1} \pi_0(\boldsymbol{V}_k)\delta(\ldots)_{\gamma=1} \quad (4.2.19)$$

where we have represented the Dirac delta term in (4.2.17) with $a \geq 0$ examples by $\delta(\ldots)_{\gamma=a}$. The self consistent equation for π_0 tells us that one solution for π_0 is simply a delta function. This is to be expected since on a regular graph under the cavity approximation for zero examples, all incoming messages will be equal to covariance matrices of a vertex on a regular tree that has seen no examples and whose neighbours up to a distance p away have also seen no examples. Denoting this covariance matrix by $\boldsymbol{V}^{(\infty)}$ we have

$$\pi_0(\boldsymbol{V}) = \delta(\boldsymbol{V} - \boldsymbol{V}^{(\infty)}) \quad (4.2.20)$$

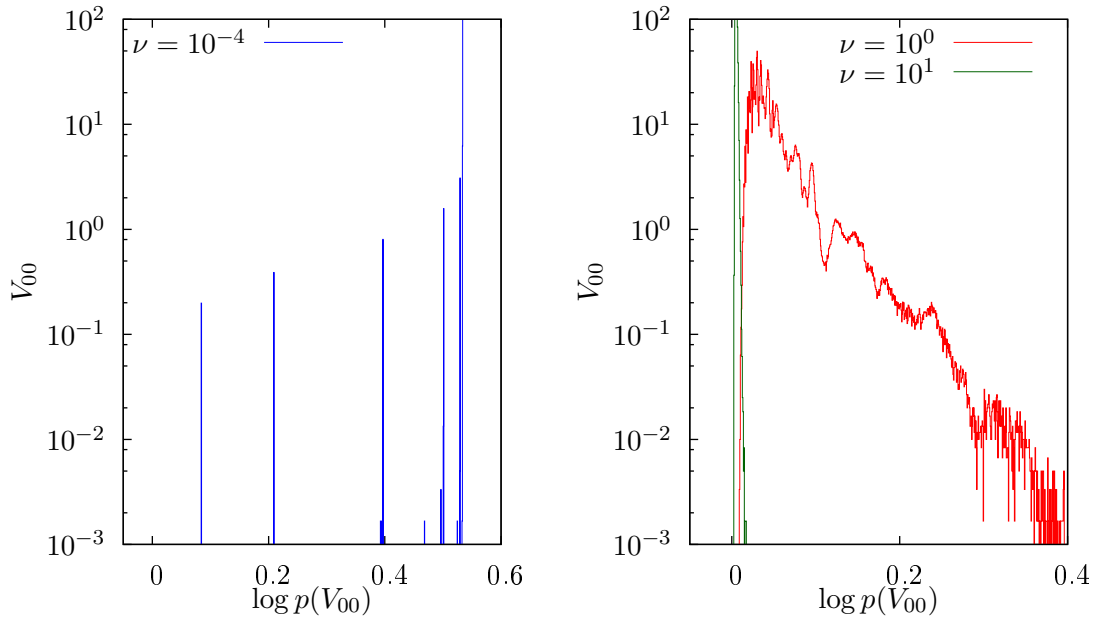
Since the initial departure from π_0 will occur when a single example appears somewhere in the vicinity of a vertex on a regular tree up to a distance p away, we expect π_1 to be a sum of Dirac deltas representing covariance matrices for vertices where an example has been seen somewhere in its a-th shell for $a = 0, \ldots, p-1$ (we expect only up to the $p-1$-th shell because a message sent with an example seen in the p-th shell is eliminated

by $\boldsymbol{XVX}$). If we represent the covariance of a vertex with an example in the a-th shell by $\boldsymbol{V}^{(a)}$ then we see that the self consistent equation for π_1 is solved by

$$\pi_1(\boldsymbol{V}) = \sum_{a=0}^{p-1} (d-1)^a \left[\delta(\boldsymbol{V} - \boldsymbol{V}^{(a)}) - \delta(\boldsymbol{V} - \boldsymbol{V}^{(0)}) \right] \tag{4.2.21}$$

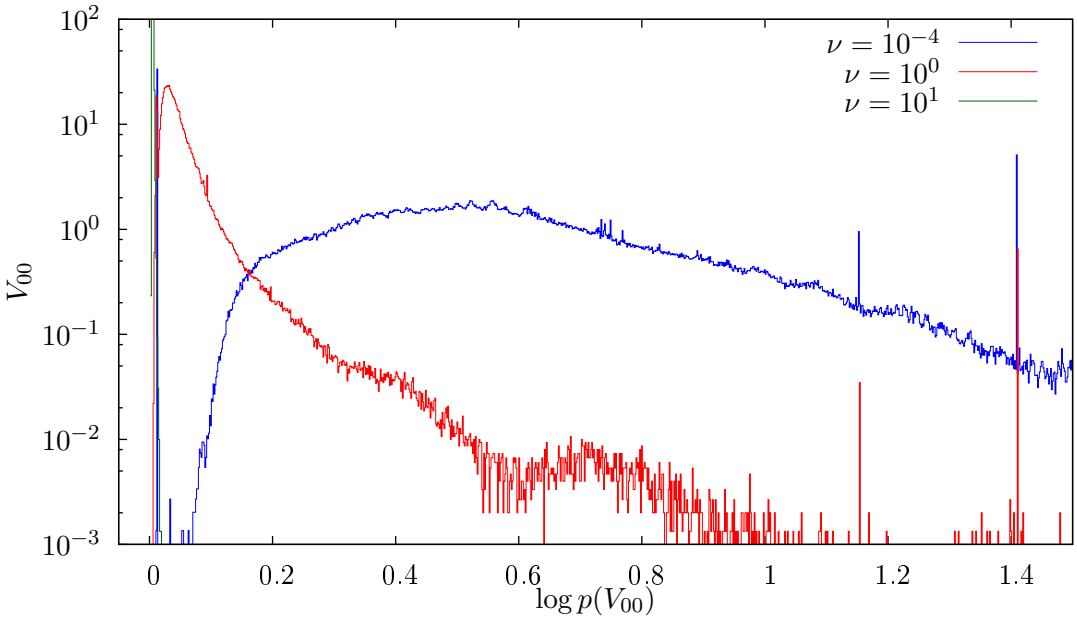
The distributions $\pi_0(\boldsymbol{V})$ and $\pi_1(\boldsymbol{V})$ explain the delta peak structure observed in figure 4.2(a). What's more, the differences in heights of the individual peaks are also accurately described by the differences in the coefficients of the delta peaks in π_1 and π_0. Figure 4.2(a) also shows, as expected, that $V_{00}^{(a)}$ decreases as a decreases. This decrease occurs because the nearer to an example a vertex is, the more the local posterior variance is reduced. For larger ν more and more peaks are added until the distribution of V_{00} becomes effectively a continuous distribution, as seen for ν equal to one. Once ν becomes very large the GP has effectively learned the target function with very little remaining uncertainty, and so the distribution of posterior variances V_{00} becomes increasingly peaked near zero; this trend can be seen for $\nu = 10$ in Figure 4.2(b).

Figure 4.2(c) shows the equivalent plot for Erdős-Rényi random graphs with average degree set to three. Since graph structure is now no longer uniform, one does not see a pattern of delta peaks for small ν as was the case for regular graphs. There are however visible peaks, around V_{00} equal to $0, 1.15$ and 1.4, superimposed on a continuous distribution. This is from vertices with degree one, three and two respectively. Since vertices with degree one have no incoming messages, all messages sent from such vertices will be identical, resulting in a large peak in the distribution of V_{00}. The same reason also gives peaks from vertices with higher degree, so long as ν is small so that the majority of vertices has not yet seen an example (corresponding to γ equal to zero in (4.2.13)). For example, the second peak in the figure is from vertices with degree three that have received two incoming messages from neighbours with degree one. As these incoming messages are deterministic, so is the outgoing message if no local example has been seen. The third peak arises similarly from degree two vertices with an incoming message from a vertex with degree one.



(a) Regular

(b) Regular



(c) Erdős-Rényi

Figure 4.2: (a) Normalised histograms of the top left entries V_{00} from a population of covariance matrices with $p = 10$ and $a = 2$ that solves the self-consistency condition (4.2.13), for random regular graphs with degree equal to three and $\nu = 0.0001$. (b) As (a) but for $\nu = 1$ and $\nu = 10$, the latter just visible as a narrow peak at the left edge of the distribution for $\nu = 1$. (c) Analogue of figures (a) and (b) combined for Erdős-Rényi random graphs with average connectivity equal to three.

4.2.2.1 Predicting prior variances

As a by-product of the cavity analysis for globally normalised kernels we note that in the cavity form of the Bayes error in equation (4.2.16), the fraction $(\gamma/\sigma^2 + d\kappa(\boldsymbol{M}_d^{-1})_{00})^{-1}$ is the local Bayes error, i.e. the local posterior variance. By keeping track of individual samples for this quantity from the population dynamics approach, we can thus predict the distribution of local posterior variances. If we set ν equal to zero, then this becomes the distribution of prior variances. The cavity approach therefore gives us, without additional effort, a prediction for this distribution.

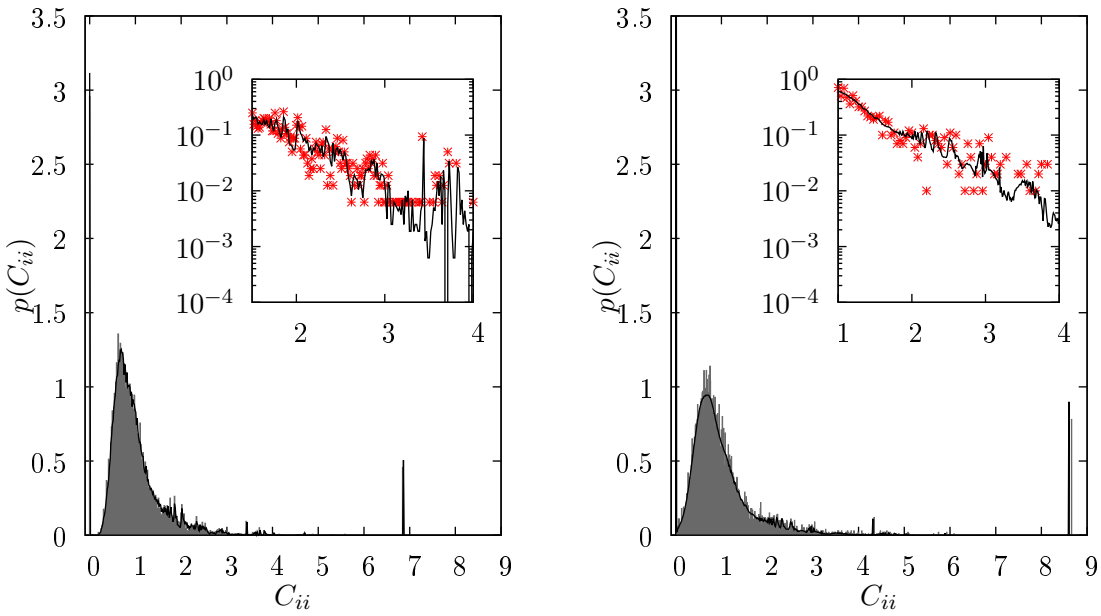
We can now go back to section 3.2 and compare the cavity predictions to numerically simulated distributions of prior variances. The cavity predictions for these distributions are shown by the black lines in figure 4.3. The cavity approach provides, in particular, detailed information about the tail of the distributions as shown in the insets. There is good agreement between the predictions and the numerical simulations, both regarding the general shape of the variance distributions and the fine structure with a number of non-trivial peaks and troughs. The residual small shifts between the predictions and the numerical results for a single instance of a large graph are most likely due to finite size effects: in a finite graph, the assumption of a tree-like local structure is not exact because there can be rare short cycles; also, the long cycles that the cavity method ignores because their length diverges logarithmically with V will have an effect when V is finite.

4.2.3 Local normalisation

With the globally normalised case under out belt we now extend the cavity analysis for the learning curves to the case of locally normalised random walk kernels, which, as argued in Chapter 3, provide more plausible probabilistic models. In this case the diagonal entries of the normalisation matrix $\boldsymbol{\mathcal{K}}$ are defined as

$$\kappa_i = \int \mathrm{d}\boldsymbol{f}\, f_i^2 P(\boldsymbol{f}), \tag{4.2.22}$$

where $P(\boldsymbol{f})$ is the GP prior with the unnormalised kernel $\hat{\boldsymbol{C}}$. This makes clear why the locally normalised kernel case is more challenging technically: we cannot calculate the normalisation constants once and for all for a given random graph ensemble and set of



(a) Erdős-Rényi

(b) Power law

Figure 4.3: (a) Grey: histogram of prior variances for the globally normalised random walk kernel with $a = 2$, $p = 10$ on a single instance of an Erdős-Rényi graph with mean degree 3 and 10000 vertices. Black: prediction for this distribution in the large graph limit. Inset: Linear-log plot of the tail of the distribution. (b) As (a) but for a power law generalised random graph with exponent 2.5 and cutoff 2.

kernel parameters p and a as we did for κ in the globally normalised scenario. Instead we have to account for the dependence of the κ_i on the specific graph instance.

On a single graph instance, this stumbling block can be overcome as follows. One iterates the cavity updates (4.2.9) for the unnormalised kernel and without the training data (i.e. setting $\kappa = 1$ and $\gamma_i = 0$). The local Bayes error at vertex i, given by the i-th term in the sum from (4.2.7), then gives us κ_i. Because γ_i equals zero, one has to use the Woodbury trick as before to get well-behaved expressions in the limit where the auxiliary parameter λ tends to zero, as was explained after (4.2.14).

Once the κ_i have been determined in this way, one can use them for predicting the Bayes error for the scenario we really want to study, i.e. using a locally normalised kernel and incorporating the training data. The relevant partition function is the analogue of (4.2.5) for local normalisation. Dropping the prefactors, the resulting partition function, Z, can be written as

$$Z \propto \int \prod_{q=0}^{p} \mathrm{d}\boldsymbol{h}^q \prod_{q=1}^{p} \mathrm{d}\hat{\boldsymbol{h}}^q \prod_{(ij)} \exp\left(-\mathrm{i}\sum_{q=1}(\hat{h}_i^q h_j^{q-1} + \hat{h}_j^q h_i^{q-1})\right) \times \prod_i \exp\left(-\frac{1}{2}\sum_{q=0}^{p} c_q d_i h_i^0 h_i^q - \frac{1}{2}\frac{d_i \kappa_i (h_i^0)^2}{\gamma_i/\sigma^2 + \lambda} + \mathrm{i}\sum_{q=1} d_i \hat{h}_i^q h_i^q\right), \quad (4.2.23)$$

where we have rescaled h_i^q to $d_i^{1/2}\kappa_i^{1/2}h_i^q$ and $\hat{h}_i^q$ to $d_i^{1/2}\kappa_i^{-1/2}\hat{h}_i^q$. Given that the κ_i have already been determined, this is a graphical model for which marginals can be calculated by iterating to a fixed point the equations for the cavity marginals:

$$P_j^{(i)}(\boldsymbol{h}_j, \hat{\boldsymbol{h}}_j|\boldsymbol{x}) \propto \exp\left(-\frac{1}{2}\sum_{q=0}^{p} c_q d_j h_j^0 h_j^q - \frac{1}{2}\frac{d_j \kappa_j (h_j^0)^2}{\gamma_j/\sigma^2 + \lambda} + \mathrm{i}\sum_{q=1}^{p} d_j \hat{h}_j^q h_j^q\right) \times \int \prod_{k\in\mathcal{N}(j)\setminus i} \mathrm{d}\boldsymbol{h}_k \mathrm{d}\hat{\boldsymbol{h}}_k \exp\left(-\mathrm{i}\sum_{q=1}^{p}(\hat{h}_j^q h_k^{q-1} + \hat{h}_k^q h_j^{q-1})\right) P_k^{(j)}(\boldsymbol{h}_k, \hat{\boldsymbol{h}}_k|\boldsymbol{x}) \quad (4.2.24)$$

As in section 4.2.2 these update equations are solved by cavity marginals of complex Gaussian form, and so we can simplify them to updates for the covariance matrices:

$$\boldsymbol{V}_j^{(i)} = \left(\boldsymbol{O}_j - \sum_{k\in\mathcal{N}(j)\setminus i} \boldsymbol{X}\boldsymbol{V}_k^{(j)}\boldsymbol{X}\right)^{-1}. \quad (4.2.25)$$

Here $\boldsymbol{X}$ is defined as in equation (4.2.11) and $\boldsymbol{O}_j$ is the obvious analogue of (4.2.10); specifically, κ is replaced by κ_j. Once the update equations have converged, one can calculate the Bayes error from a similarly adapted version of (4.2.7).

The above procedure for a single fixed graph now has to be extended to the case of an ensemble of large random graphs characterised by some degree distribution $p(d)$. The outcome of the first round of cavity updates, for the unnormalised kernel without training data, is then represented by a distribution of cavity covariances $\boldsymbol{V}_{\text{aux}}$, while the second one gives a distribution of cavity covariances, $\boldsymbol{V}$, for the locally normalised kernel, with training data included. Importantly, these message distributions are coupled to each other via the graph structure, so we need to look at the joint distribution $\pi(\boldsymbol{V}, \boldsymbol{V}_{\text{aux}})$.

Detailed analysis using the replica method (see Chapter 5) shows that the correct fixed point equation updates the $\boldsymbol{V}_{\text{aux}}$-messages as in the globally normalised case with γ equal to zero. The second set of local covariances, $\boldsymbol{V}$, are then updated according to (4.2.25), with a normaliser calculated using the marginals from the $d-1$ $\boldsymbol{V}_{\text{aux}}$-covariances and an additional 'counterflow' covariance generated from $\pi(\boldsymbol{V}_{\text{aux}}) = \int d\boldsymbol{V}\pi(\boldsymbol{V}, \boldsymbol{V}_{\text{aux}})$, subject to the constraint that the local marginals of the neighbours are consistent. We find in practice that the consistency constraint can be dropped and the fixed point equation for the distribution of the two sets of messages can be approximated by

$$\pi(\boldsymbol{V}, \boldsymbol{V}_{\text{aux}}) = \left\langle \sum_d \frac{p(d)d}{\bar{d}} \int \prod_{k=1}^{d-1} \mathrm{d}\boldsymbol{V}_k \mathrm{d}\boldsymbol{V}_{\text{aux},k} \mathrm{d}\boldsymbol{V}_{\text{aux},d} \prod_{k=1}^{d-1} \pi(\boldsymbol{V}_k, \boldsymbol{V}_{\text{aux},k}) \pi(\boldsymbol{V}_{\text{aux},d}) \right.$$
$$\times \delta\left(\boldsymbol{V}_{\text{aux}} - \left(\boldsymbol{O}_{\text{aux}} - \sum_{k=1}^{d-1} \boldsymbol{X}\boldsymbol{V}_{\text{aux},k}\boldsymbol{X}\right)^{-1}\right)$$
$$\left. \times\delta\left(\boldsymbol{V} - \left(\boldsymbol{O} - \sum_{k=1}^{d-1} \boldsymbol{X}\boldsymbol{V}_k\boldsymbol{X}\right)^{-1}\right)\right\rangle_{\gamma} . \quad (4.2.26)$$

with $\boldsymbol{O}_{\text{aux}}$ defined as (4.2.10) with γ equal to zero and κ equal to one. One sees that if one marginalises over $\boldsymbol{V}$, then one obtains exactly the same condition on $\pi(\boldsymbol{V}_{\text{aux}})$ as before in the globally normalised kernel case (but with $\kappa = 1$ and $\nu = 0$), see (4.2.13). This reflects the fact that the cavity updates for the first set of messages on a single graph do not rely on any information about the second set. The first delta function in (4.2.26) corresponds to the fixed point condition for this second set of cavity updates. This condition depends, via the value of the local κ, on the $\boldsymbol{V}_{\text{aux}}$-cavity covariances:

$$\kappa = \frac{1}{d(\boldsymbol{M}^{-1}_{\text{aux},d})_{00}} \quad (4.2.27)$$

where $\boldsymbol{M}_{\text{aux},d}$ is defined in a similar manner to (4.2.16) again with κ equal to one. It may seem unusual that d copies of $\boldsymbol{V}_{\text{aux}}$ enter here; $\boldsymbol{V}_{\text{aux},d}$ represents the cavity covariance

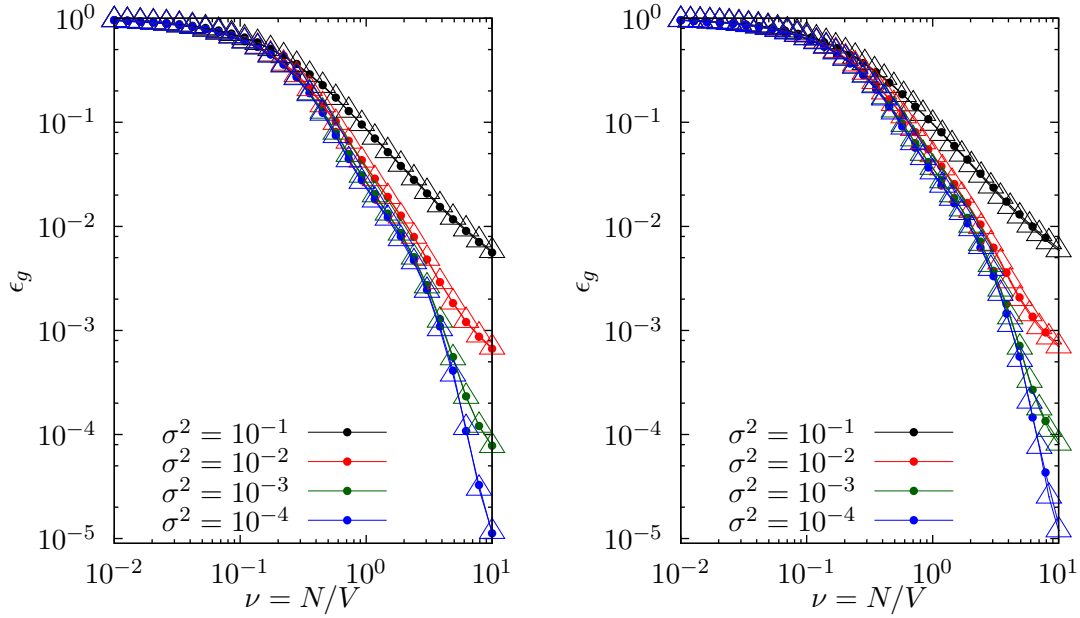
from the first set that is *received* from the vertex to which the new message $\boldsymbol{V}$ is being *sent*. While this 'counterflow' appears to run against the basic construction of the cavity or belief propagation method, it makes sense here because the first set of cavity messages (or equivalently the distribution $\pi(\boldsymbol{V}_{\text{aux}})$) reaches a fixed point that is independent of the second set, so the counterflow of information is only apparent. The reason why knowledge about $\boldsymbol{V}_{\text{aux},d}$ is needed in the update is that κ is the variance of a full marginal rather than a cavity marginal.

In a similar manner to the case of global normalisation, (4.2.26) can be solved by looking for a fixed point of $\pi(\boldsymbol{V}, \boldsymbol{V}_{\text{aux}})$ using population dynamics (see algorithm 4.1). Updates are made by first updating $\boldsymbol{V}_{\text{aux}}$ using equation (4.2.8) and then updating $\boldsymbol{V}$ using (4.2.25) with $\kappa \equiv \kappa_i$ replaced by (4.2.27).

Once a fixed point has been calculated for the covariance distribution we apply the Woodbury formula to (4.2.7) in a similar manner to section 4.2.2 to give the prediction for the learning curve for GP regression with a locally normalised kernel. The result for the Bayes error becomes

$$\epsilon = \left\langle \sum_d p(d) \int \prod_{k=1}^{d} \mathrm{d}\boldsymbol{V}_k \mathrm{d}\boldsymbol{V}_{\text{aux},k} \prod_{k=1}^{d} \pi(\boldsymbol{V}_k, \boldsymbol{V}_{\text{aux},k}) \times \frac{1}{\gamma/\sigma^2 + (\boldsymbol{M}_d^{-1})_{00}/(\boldsymbol{M}_{\text{aux},d}^{-1})_{00}} \right\rangle_\gamma . \quad (4.2.28)$$

Learning curve predictions for GPs with locally normalised kernels as they result from the cavity approach described above are shown in figure 4.4. The figure shows numerically simulated learning curves and the cavity prediction, both for Erdős-Rényi random graphs and power law generalised random graphs with five hundred vertices. As for the globally normalised case one sees that the cavity predictions are quantitatively very accurate, even with the simplified update equation (4.2.26). They capture all aspects of learning curve both qualitatively and quantitatively, including e.g. the shoulder in the curves from disconnected single vertices (see section 3.3).



(a) Erdős-Rényi

(b) Power law

Figure 4.4: (a) Learning curves for GP regression with locally normalised kernels with $p = 10$, $a = 2$ on Erdős-Rényi random graphs with mean degree equal to three, for a range of noise levels, σ^2. Solid lines with filled circles: numerically simulated learning curves for graphs with five hundred vertices, solid lines with triangles: cavity predictions. (b) As (a) for power law generalised random graphs with exponent 2.5 and cutoff 2.

4.3 Learning curves for large p

Before moving on to deriving in more detail the counterflow relationship mentioned in the cavity derivation for local kernel normalisation, we finally look at how the learning curves for GP regression on graphs depend on the kernel lengthscale p/a. We focus for this discussion on random regular graphs, where the distinction between global and local normalisation is not important. In section 3.1.1, we saw that on a large regular graph the random walk kernel approaches a non-trivial limiting form for large p, as long as one stays below the threshold (3.1.5) for p where cycles become important. One might be tempted to conclude from this that also the learning curves have a limiting form for large p. This is too naïve, however, as one can see by considering e.g. the effect of the first example on the Bayes error. If the example is at vertex i, the posterior variance at vertex j is, from (2.1.9), $C_{jj} - C_{ij}^2/(C_{ii}+\sigma^2)$. As the prior variances C_{jj} are all equal, to unity for our chosen normalisation, this is $1 - C_{ij}^2/(1+\sigma^2)$. The reduction in Bayes error is therefore $\epsilon(0) - \epsilon(1) = (1/V)\sum_j C_{ij}^2/(1+\sigma^2)$. As long as cycles are unimportant this is independent of the location of the example vertex i, and in the notation of section 3.1.1 can be written as

$$\epsilon(0) - \epsilon(1) = \frac{1}{1+\sigma^2} \sum_{l=0}^{p} v_l C_{l,p}^2 \tag{4.3.1}$$

where v_l, as before, is the number of vertices a distance l away from vertex i, i.e. v_0 equal to one and $v_l = d(d-1)^{l-1}$ for $l \geq 1$. To evaluate (4.3.1) for large p one cannot directly plug in the limiting kernel form (3.1.3): the 'shell volume' v_l just balances the l-dependence of the factor $(d-1)^{-l/2}$ from $C_{l,p}$, so that one gets contributions from all distances l, proportional to l^2 for large l. Naïvely summing up to $l = p$ would give an initial decrease of the Bayes error growing as p^3. This is not correct; the reason is that while $\hat{C}_{l,p}$ approaches the large p-limit (3.1.3) for any fixed l, it does so more and more slowly as l increases. A more detailed analysis (see Appendix A) shows that for large l and p, $\hat{C}_{l,p}$ is proportional to the large p-limit $l(d-1)^{-l/2}$ up to a characteristic cut off distance l of order $p^{1/2}$, and decays quickly beyond this. Summing in (4.3.1) the contributions of order l^2 up to this distance predicts finally that the initial error decay should scale, non-trivially, as $p^{3/2}$.

We next show that this large p-scaling with $p^{3/2}$ is also predicted, for the entire learning curve, by the eigenvalue approximation (2.2.38). As before we consider d-regular random

graphs. The required spectrum of kernel eigenvalues λ_α becomes identical, for large V, to that on a d-regular tree [75]. Explicitly, if λ_α^L are the eigenvalues of the normalised graph Laplacian on the tree, then the kernel eigenvalues are $\lambda_\alpha = \kappa^{-1}V^{-1}(1-\lambda_\alpha^L/a)^p$. Here the factor V^{-1} comes from the same factor in the kernel eigenvalue definition after (2.2.38), and κ is the overall normalisation constant which enforces $\sum_\alpha \lambda_\alpha = V^{-1}\sum_j C_{jj} = 1$. The spectrum of the Laplacian on a regular tree is known [25, 75] and is given by

$$\rho(\lambda^L) = \begin{cases} \frac{\sqrt{\frac{4(d-1)}{d^2} - (\lambda^L - 1)^2}}{(2\pi/d)\lambda^L(2-\lambda^L)} & \lambda_- \leq \lambda \leq \lambda_+ \\ 0 & \text{otherwise} \end{cases} \tag{4.3.2}$$

where $\lambda_\pm = 1 \pm \frac{2}{d}(d-1)^{1/2}$. (There are also two isolated eigenvalues at zero and two, which do not contribute for large V.)

We can now write down the function g from (2.2.38), converting the sum over kernel eigenvalues to V times an integral over Laplacian eigenvalues for large V. Dropping the L superscript, the result is

$$g(h) = \int_{\lambda_-}^{\lambda_+} \mathrm{d}\lambda\, \rho(\lambda)[\kappa(1-\lambda/a)^{-p} + hV^{-1}]^{-1} \tag{4.3.3}$$

The dependence on hV^{-1} here shows that in the approximate learning curve (2.2.38), the Bayes error depends only on $\nu = N/V$, in agreement with our cavity method results. The condition for the normalisation factor κ becomes simply $g(0) = 1$, or $\kappa^{-1} = \int \mathrm{d}\lambda\, \rho(\lambda)(1-\lambda/a)^p$.

So far we have written down how one would evaluate the eigenvalue approximation to the learning curve on large d-regular random graphs, for arbitrary kernel parameters p and a. Now we want to consider the large p-limit. We show that there is then a *master curve* for the Bayes error against $\nu p^{3/2}$. This is entirely consistent with the $p^{3/2}$ scaling found above for the initial error decay. The intuition for the large p analysis is that the factor $(1-\lambda/a)^p$ decays quickly as the Laplacian eigenvalue λ increases beyond λ_-, so that only values of λ near λ_- contribute. One can then approximate

$$\left(1 - \frac{\lambda}{a}\right)^p \approx \left(1 - \frac{\lambda_-}{a}\right)^p \exp\left(-\frac{p(\lambda - \lambda_-)}{a - \lambda_-}\right) \tag{4.3.4}$$

Similarly one can replace $\rho(\lambda)$ by its leading square root behaviour near λ_-,

$$\rho(\lambda) = (\lambda - \lambda_-)^{1/2} r \tag{4.3.5}$$

where $r = (d-1)^{1/4} d^{5/2}/(\pi(d-2)^2)$. Substituting these approximations into (4.3.3) and introducing the rescaled integration variable $y = p(\lambda - \lambda_-)/(a - \lambda_-)$ gives

$$g(h) = r\kappa^{-1}(1-\lambda_-/a)^p \left(\frac{a-\lambda_-}{p}\right)^{3/2} F(h\kappa^{-1}V^{-1}(1-\lambda_-/a)^p), \tag{4.3.6}$$

with $F(z) = \int_0^\infty \mathrm{d}y\, y^{1/2}(\exp(y)+z)^{-1}$. Since $g(0) = 1$, the prefactor must equal $1/F(0) = 2/\sqrt{\pi}$. This fixes the normalisation constant κ, and we can simplify to

$$g(h) = \frac{F(hV^{-1}c^{-1})}{F(0)}, \qquad c = rF(0)\left(\frac{a-\lambda_-}{p}\right)^{3/2}. \tag{4.3.7}$$

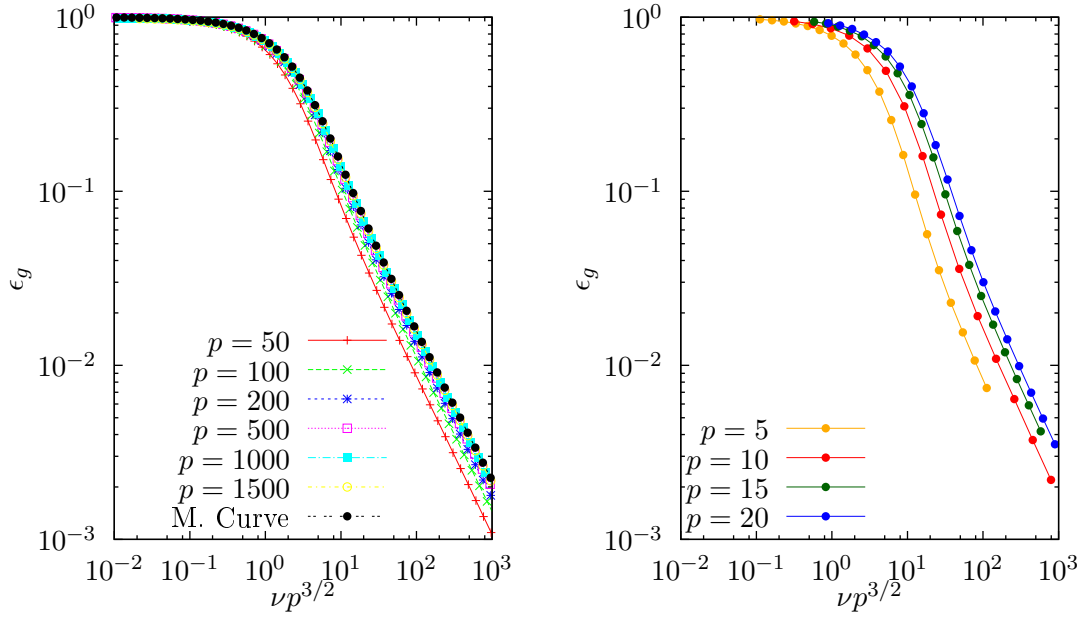
The learning curves for large p are then predicted from (2.2.38) by solving

$$\epsilon = F(\nu c^{-1}/(\epsilon+\sigma^2))/F(0) \tag{4.3.8}$$

and depend clearly only on the combination νc^{-1}. Because c is proportional to $p^{-3/2}$, this shows that learning curves for different p should collapse onto a master curve when plotted against $\nu p^{3/2}$.

A plot of the scaling of the eigenvalue learning curve approximations onto the master curve is shown in figure 4.5(a). As can be seen, large values of p are required in order to get a good collapse in the tail of the learning curve prediction, whereas in the initial part the $p^{3/2}$ scaling is accurate already for relatively small p.

Finally, figure 4.5(b) shows that the predicted $p^{3/2}$-scaling of the learning curves is present not only within the eigenvalue approximation, but also in the actual learning curves. Figure 4.5(b) displays numerically simulated learning curves for $p = 5, 10, 15, 20$, against the rescaled number of examples $\nu p^{3/2}$ as before. Even for these comparatively small values of p one sees that the rescaled learning curves begin to approach a master curve.



(a) Eigenvalue approximation

(b) Numerics

Figure 4.5: (a) Eigenvalue approximation for learning curves on a random 3-regular graph, using a random walk kernel with $a = 2$, $\sigma^2 = 0.1$ and increasing values of p as shown. Plotting against $\nu p^{3/2}$ shows that for large p these rescaled curves approach the master curve predicted from (4.3.8), though this approach is slower in the tail of the curves. (a) As (b), but for numerically simulated learning curves on graphs with five hundred vertices

4.4 Summary

In this chapter we have shown that by using the cavity method presented in section 2.3 we may accurately predict the learning curves of a GP with a random walk kernel in a matched scenario where the teacher and student GP have the same hyperparameters (see section 2.1.4.2). We began by first considering the case of global normalisation where the covariance function or kernel of the GP prior is normalised so that the average prior variance of the GP was constant. We showed that we were able to rewrite this in terms of a graphical model so that the cavity method could be applied. We saw in figure 4.1 that this gave predictions that were indistinguishable from numerics, and we expect that in the large vertex limit the predictions will in the end become exact.

Once we had calculated the learning curves for the case of a globally normalised prior we then looked at calculating the learning curves of a GP with a locally normalised kernel, where the prior scale of the function is fixed to one for all vertices. We showed that this presented a more technically difficult problem. We found that, although the specific graph instance BP equations were simple to write down (see (4.2.24)), the cavity method was more complicated. It required an additional approximation that amounted to ignoring consistency constraints between incoming cavities. We found that even after ignoring these constraints the cavity prediction was still accurate and again indistinguishable from numerics, surprisingly so given the approximation.

Finally we looked at the scaling of the learning curves of a GP with a globally normalised random walk kernel. We saw that by using the baseline prediction derived in section 2.2 on random d-regular graphs where the spectrum of the normalised Laplacian is known in the large vertex limit we were able to derive a master curve. We showed that for large lengthscales learning curve predictions could be scaled onto a master curve by rescaling the number of examples according to $\nu p^{3/2}$. In figure 4.5(a) we plotted this scaling of the learning curve and showed it required large kernel lengthscales before it became accurate. To check that this scaling was present in the real GP and not just the baseline comparison we concluded in figure 4.5(b) by plotting rescaled numerically calculated learning curves. We found that, contrary to the baseline comparison, relatively small lengthscales began to scale onto a master curve.

Now that we have derived a prediction of the learning curve for GP regression in a matched scenario with a random walk kernel we will investigate the approximation we made in deriving (4.2.26). We will see that by applying the replica method this will lead to some interesting and non-trivial problems that must be solved.

CHAPTER 5

UNDERSTANDING COUNTERFLOW – THE REPLICA METHOD

Whilst calculating the generalisation error of GPs for regression on graphs in the previous chapter we came across a rather counter-intuitive result for the case of a locally normalised kernel. In particular we found that each update requires a counterflow of information in order to calculate an auxiliary marginal distribution. In this chapter we will derive the same result using the replica method. By approximating the generalisation error in this way the origin of the counterflow of information will become clear. The replica derivation will also enable us to discuss in more detail the approximation leading to the self consistent equation given in (4.2.26). The derivation in this chapter will be similar in approach to that of Kühn [64] but requiring significant extensions for locally normalised kernels. We begin with a replica derivation of the generalisation error for a GP prior with a globally normalised kernel. As in the cavity analysis this will serve as a useful warm up to the more complicated locally normalised kernel.

5.1 Global normalisation

We pick up from the cavity approximation of the generalisation error in Chapter 4 after (4.2.6). We wish to include the average over the disorder in the partition function. Using the replica trick

$$\langle \log Z \rangle_{\{\gamma_i\},\mathcal{G}} = \lim_{n\to\infty} \frac{1}{n} \log \langle Z^n \rangle_{\{\gamma_i\},\mathcal{G}} \tag{5.1.1}$$

we see that we must consider an n-replicated version of (4.2.6), averaged over the graph ensemble, $\mathcal{G}$, and the vertex example distribution, $\gamma_i \sim \text{Poisson}(\nu)$. The n-replicated partition function will therefore be given by

$$\langle Z^n \rangle_{\{\gamma_i\},\mathcal{G}} = \left\langle \int \prod_{a=1}^{n} \prod_{i=1}^{V} \mathrm{d}\boldsymbol{r}_i^a \prod_{i=1}^{V} \exp\left(-\sum_{a=1}^{n} \mathcal{H}(\boldsymbol{r}_i^a, d_i, \kappa, \gamma_i) \right) \times \prod_{(i,j)\in\mathcal{E}} \exp\left(-\sum_{a=1}^{n} \mathcal{J}(\boldsymbol{r}_i^a, \boldsymbol{r}_j^a) \right) \right\rangle_{\{\gamma_i\},\mathcal{G}} . \tag{5.1.2}$$

where, for notational simplicity, we have denoted the vector of all of a vertex's local variables by $\boldsymbol{r}_i = (h_i^0, \ldots, h_i^p, \hat{h}_i^1, \ldots, \hat{h}_i^p)^{\mathrm{T}}$ and defined the functions

$$\mathcal{H}(\boldsymbol{r}, d, \kappa, \gamma) = \frac{d}{2} \sum_{q=0}^{p} c_q h^0 h^q + \frac{\kappa d (h^0)^2}{2(\gamma/\sigma^2 + \lambda)} - \mathrm{i} d \sum_{q=1}^{p} \hat{h}^q h^q - \frac{1}{2} \log\left(\frac{1}{\gamma/\sigma^2 + \lambda} \right) \tag{5.1.3}$$

$$\mathcal{J}(\boldsymbol{r}, \boldsymbol{r}') = \mathrm{i} \sum_{q=1}^{p} \left(\hat{h}^q (h')^{q-1} + (\hat{h}')^q h^{q-1} \right) . \tag{5.1.4}$$

We consider graph ensembles with probability distributions of the form

$$P(G|\{d_1, \ldots, d_V\}) = \frac{1}{\mathcal{N}} \prod_{i<j} p(a_{ij}) \delta_{a_{ij}, a_{ji}} \prod_{i=1}^{V} \delta_{d_i, \sum_{j\neq i} a_{ij}} \tag{5.1.5}$$

$$p(a_{ij}) = \frac{\bar{d}}{V} a_{ij} + \left(1 - \frac{\bar{d}}{V} \right) (1 - a_{ij}) \tag{5.1.6}$$

where $\mathcal{N}$ is the normalising constant, $\{d_1, \ldots, d_V\}$ is a sequence sampled i.i.d. from a degree distribution $p(d)$ and a_{ij} is the ij-th element of the adjacency matrix, $\boldsymbol{A}$, which is one when there is an edge between i and j and zero otherwise. This distribution selects all graphs with the prescribed degree sequence uniformly from the space of all graphs, and encompasses all the graph ensembles discussed in section 2.4 [see e.g. 23, 92, 118].

Explicitly averaging over the graph ensemble we have

$$\langle Z^n\rangle_{\{\gamma_i\},\mathcal{G}} = \frac{1}{\mathcal{N}}\left\langle \int \prod_{a=1}^{n}\prod_{i=1}^{V} \mathrm{d}\boldsymbol{r}_i^a \prod_{i=1}^{V} \exp\left(-\sum_{a=1}^{n}\mathcal{H}(\boldsymbol{r}_i^a, d_i, \kappa, \gamma_i)\right) \right.$$
$$\left. \times \sum_{\{a_{ij}\}}\prod_{i<j}\exp\left(-\sum_{a=1}^{n} a_{ij}\mathcal{J}(\boldsymbol{r}_i^a, \boldsymbol{r}_j^a)\right)\prod_{i<j} p(a_{ij})\delta_{a_{ij},a_{ji}} \prod_{i=1}^{V}\delta_{d_i,\sum_{j\neq i}a_{ij}}\right\rangle_{\{\gamma_i\}} \tag{5.1.7}$$

which upon rewriting the Kronecker delta degree constraint in its Fourier form then gives

$$\langle Z^n\rangle_{\{\gamma_i\},\mathcal{G}} = \frac{1}{\mathcal{N}}\left\langle \int \prod_{a=1}^{n}\prod_{i=1}^{V} \mathrm{d}\boldsymbol{r}_i^a \prod_{i=1}^{V} \exp\left(-\sum_{a=1}^{n}\mathcal{H}(\boldsymbol{r}_i^a, d_i, \kappa, \gamma_i)\right) \right.$$
$$\times \sum_{\{a_{ij}\}}\prod_{i<j}\exp\left(-\sum_{a=1}^{n} a_{ij}\mathcal{J}(\boldsymbol{r}_i^a, \boldsymbol{r}_j^a)\right) p(a_{ij})\delta_{a_{ij},a_{ji}}$$
$$\left. \times \prod_{i=1}^{V}\int_0^{2\pi}\frac{\mathrm{d}\hat{d}_i}{2\pi}\mathrm{e}^{-\mathrm{i}\hat{d}_i(d_i-\sum_{i\neq j}a_{ij})}\right\rangle_{\{\gamma_i\}}, \tag{5.1.8}$$

Using a change of dummy index in the final exponent we see that (5.1.8) can be simplified and becomes

$$\langle Z^n\rangle_{\{\gamma_i\},\mathcal{G}} = \frac{1}{\mathcal{N}}\left\langle \int_0^{2\pi}\prod_{i=1}^{V}\frac{\mathrm{d}\hat{d}_i}{2\pi}\int\prod_{a=1}^{n}\prod_{i=1}^{V}\mathrm{d}\boldsymbol{r}_i^a \right.$$
$$\times \sum_{\{a_{ij}\}}\prod_{i<j}\exp\left(-\sum_{a=1}^{n} a_{ij}\mathcal{J}(\boldsymbol{r}_i^a, \boldsymbol{r}_j^a) + \mathrm{i}a_{ij}(\hat{d}_i+\hat{d}_j)\right) p(a_{ij})$$
$$\left. \times \prod_{i=1}^{V}\exp\left(-\sum_{a=1}^{n}\mathcal{H}(\boldsymbol{r}_i^a, d_i, \kappa\gamma_i) - \mathrm{i}\hat{d}_i d_i\right)\right\rangle_{\{\gamma_i\}}. \tag{5.1.9}$$

If we now explicitly put the form of $p(a_{ij})$ given in (5.1.6) into (5.1.9) we can sum over the $\{a_{ij}\}$ and are left with

$$\langle Z^n\rangle_{\{\gamma_i\},\mathcal{G}} = \frac{1}{\mathcal{N}}\left\langle \int_0^{2\pi}\prod_{i=1}^{V}\frac{\mathrm{d}\hat{d}_i}{2\pi}\int\prod_{a=1}^{n}\prod_{i=1}^{V}\mathrm{d}\boldsymbol{r}_i^a \right.$$
$$\times \prod_{i<j}\left[\frac{\bar{d}}{V}\exp\left(-\sum_{a=1}^{n}\mathcal{J}(\boldsymbol{r}_i^a, \boldsymbol{r}_j^a) + \mathrm{i}(\hat{d}_i+\hat{d}_j)\right) + \left(1-\frac{\bar{d}}{V}\right)\right]$$
$$\left. \times \prod_{i=1}^{V}\exp\left(-\sum_{a=1}^{n}\mathcal{H}(\boldsymbol{r}_i^a, d_i, \kappa, \gamma_i) - \mathrm{i}\hat{d}_i d_i\right)\right\rangle_{\{\gamma_i\}} \tag{5.1.10}$$

which can be simplified further by making the large V approximation, $\exp(f(\boldsymbol{r})/V) \approx 1+f(\boldsymbol{r})/V$ for some function f that does not grow with V, in the second line. Performing

this simplification we finally have,

$$\langle Z^n\rangle_{\{\gamma_i\},\mathcal{G}} = \frac{1}{\mathcal{N}}\left\langle \int_0^{2\pi}\prod_{i=1}^{V}\frac{\mathrm{d}\hat{d}_i}{2\pi}\int\prod_{a=1}^{n}\prod_{i=1}^{V}\mathrm{d}\boldsymbol{r}_i^a \times\prod_{i,j}\exp\left(\frac{\bar{d}}{2V}\left[\exp\left(-\sum_{a=1}^{n}\mathcal{J}(\boldsymbol{r}_i^a,\boldsymbol{r}_j^a)+\mathrm{i}(\hat{d}_i+\hat{d}_j)\right)-1\right]\right) \times\prod_{i=1}^{V}\exp\left(-\sum_{a=1}^{n}\mathcal{H}(\boldsymbol{r}_i^a,d_i,\kappa,\gamma_i)-\mathrm{i}\hat{d}_i d_i\right)\right\rangle_{\{\gamma_i\}} \tag{5.1.11}$$

5.1.1 Deriving the 'effective single site' formulation

As is typical for a replica calculation we aim to rewrite (5.1.11) in an 'effective single site' formulation, i.e. we want to express (5.1.11) in a form without the indices i and j indicating specific vertices in the graph. We do this by rewriting (5.1.2) in terms of 'replica densities'. We will define these replica densities as

$$\rho(\boldsymbol{r}^1,\ldots,\boldsymbol{r}^n)=\frac{1}{V}\sum_{i=1}^{V}\prod_{a=1}^{n}\delta(\boldsymbol{r}^a-\boldsymbol{r}_i^a)\mathrm{e}^{\mathrm{i}\hat{d}_i}, \tag{5.1.12}$$

and incorporate them into (5.1.2) with a functional delta using conjugate densities $\hat{\rho}$ via

$$\begin{aligned}1&=\int\mathcal{D}\rho\,\delta\left(V\rho(\ldots)-\sum_{i=1}^{V}\prod_{a=1}^{n}\delta(\boldsymbol{r}^a-\boldsymbol{r}_i^a)\mathrm{e}^{\mathrm{i}\hat{d}_i}\right)\\ &=\int\mathcal{D}\rho\int\mathcal{D}\hat{\rho}\,\exp\left(\mathrm{i}\int\prod_a\mathrm{d}\boldsymbol{r}^a\hat{\rho}(\ldots)\left(\sum_{i=1}^{V}\prod_{a=1}^{n}\delta(\boldsymbol{r}^a-\boldsymbol{r}_i^a)\mathrm{e}^{\mathrm{i}\hat{d}_i}-V\rho(\ldots)\right)\right).\end{aligned} \tag{5.1.13}$$

Here in order to shorten the above equations we have abbreviated $\rho(\boldsymbol{r}^1,\ldots,\boldsymbol{r}^n)$ by $\rho(\ldots)$.

Substituting (5.1.13) into (5.1.11) we have

$$\langle Z^n\rangle_{\{\gamma_i\},\mathcal{G}} = \frac{1}{\mathcal{N}}\left\langle \int\mathcal{D}\rho\mathcal{D}\hat{\rho}\int_0^{2\pi}\prod_{i=1}^{V}\frac{\mathrm{d}\hat{d}_i}{2\pi}\int\prod_{a=1}^{n}\prod_{i=1}^{V}\mathrm{d}\boldsymbol{r}_i^a \times\exp\left(\mathrm{i}\int\prod_a\mathrm{d}\boldsymbol{r}^a\hat{\rho}(\ldots)\left(\sum_{i=1}^{V}\prod_{a=1}^{n}\delta(\boldsymbol{r}^a-\boldsymbol{r}_i^a)\mathrm{e}^{\mathrm{i}\hat{d}_i}-V\rho(\ldots)\right)\right) \times\prod_{i,j}\exp\left(\frac{\bar{d}}{2V}\left[\exp\left(-\sum_{a=1}^{n}\mathcal{J}(\boldsymbol{r}_i^a,\boldsymbol{r}_j^a)+\mathrm{i}(\hat{d}_i+\hat{d}_j)\right)-1\right]\right) \times\prod_{i=1}^{V}\exp\left(-\sum_{a=1}^{n}\mathcal{H}(\boldsymbol{r}_i^a,d_i,\kappa,\gamma_i)-\mathrm{i}\hat{d}_i d_i\right)\right\rangle_{\{\gamma_i\}}. \tag{5.1.14}$$

From (5.1.14) we see that the $\hat{\rho}(\ldots)\rho(\ldots)$ is immediately in a form independent of specific vertices. Furthermore, by rewriting the $\mathcal{J}$ term in terms of $\rho(\ldots)$ we can also eliminate the vertex dependence in the $\mathcal{J}$ term. This leaves us with

$$\begin{aligned}\langle Z^n\rangle_{\{\gamma_i\},\mathcal{G}} = \frac{1}{\mathcal{N}}\int \mathcal{D}\rho\mathcal{D}\hat{\rho}\exp\left(\frac{V\bar{d}}{2}\left(S_1[\rho]-1\right)-\mathrm{i}VS_2[\rho,\hat{\rho}]\right)\\ \times\left\langle\int_0^{2\pi}\prod_{i=1}^{V}\frac{\mathrm{d}\hat{d}_i}{2\pi}\int\prod_{a=1}^{n}\prod_{i=1}^{V}\mathrm{d}\boldsymbol{r}_i^a\right.\\ \left.\prod_{i=1}^{V}\exp\left(\mathrm{i}\hat{\rho}(\ldots_i)\mathrm{e}^{\mathrm{i}\hat{d}_i}-\sum_{a=1}^{n}\mathcal{H}(\boldsymbol{r}_i^a,d_i,\kappa,\gamma_i)-\mathrm{i}\hat{d}_i d_i\right)\right\rangle_{\{\gamma_i\}}\end{aligned}\tag{5.1.15}$$

where we have defined

$$\begin{aligned}S_1[\rho] = \int\prod_{a=1}^{n}\mathrm{d}\boldsymbol{r}^a\mathrm{d}(\boldsymbol{r}^a)'\rho(\boldsymbol{r}^1,\ldots,\boldsymbol{r}^n)\rho((\boldsymbol{r}^1)',\ldots,(\boldsymbol{r}^n)')\\ \times\exp\left[-\sum_{a=1}^{n}\mathcal{J}(\boldsymbol{r}^a,(\boldsymbol{r}^a)')\right]\end{aligned}\tag{5.1.16}$$

$$S_2[\rho,\hat{\rho}] = \int\prod_{a=1}^{n}\mathrm{d}\boldsymbol{r}^a\rho(\boldsymbol{r}^1,\ldots,\boldsymbol{r}^n)\hat{\rho}(\boldsymbol{r}^1,\ldots,\boldsymbol{r}^n).\tag{5.1.17}$$

With $\langle Z^n\rangle_{\{\gamma_i\},\mathcal{G}}$ in the form (5.1.15) all that remains to achieve the desired effective single site form is to deal with the last line of (5.1.15). Some progress can be made by rewriting the last line in terms of an exponential of a logarithm to give

$$\begin{aligned}&\left\langle\int_0^{2\pi}\prod_{i=1}^{V}\frac{\mathrm{d}\hat{d}_i}{2\pi}\int\prod_{a=1}^{n}\prod_{i=1}^{V}\mathrm{d}\boldsymbol{r}_i^a\prod_{i=1}^{V}\exp\left(\mathrm{i}\hat{\rho}(\ldots_i)\prod_{a=1}^{n}\mathrm{e}^{\mathrm{i}\hat{d}_i}-\sum_{a=1}^{n}\mathcal{H}(\boldsymbol{r}_i^a,d_i,\kappa,\gamma_i)-\mathrm{i}\hat{d}_i d_i\right)\right\rangle_{\{\gamma_i\}}\\ &=\exp\left(\sum_{i=1}^{V}\log\left\langle\int_0^{2\pi}\frac{\mathrm{d}\hat{d}_i}{2\pi}\int\prod_{a=1}^{n}\mathrm{d}\boldsymbol{r}_i^a\exp\left(\mathrm{i}\hat{\rho}(\ldots_i)\mathrm{e}^{\mathrm{i}\hat{d}_i}-\sum_{a=1}^{n}\mathcal{H}(\boldsymbol{r}_i^a,d_i,\kappa,\gamma_i)-\mathrm{i}\hat{d}_i d_i\right)\right\rangle_{\gamma_i}\right).\end{aligned}$$

By rewriting $\exp\left(\mathrm{i}\hat{\rho}(\ldots_i)\prod_{a=1}^{n}\mathrm{e}^{\mathrm{i}\hat{d}_i}\right)$ in terms of its power series and rearranging terms this can further be rewritten and we see that

$$\begin{aligned}&\left\langle\int_0^{2\pi}\prod_{i=1}^{V}\frac{\mathrm{d}\hat{d}_i}{2\pi}\prod_{a=1}^{n}\prod_{i=1}^{V}\mathrm{d}\boldsymbol{r}_i^a\prod_{i=1}^{V}\exp\left(\mathrm{i}\hat{\rho}(\ldots_i)\prod_{a=1}^{n}\mathrm{e}^{\mathrm{i}\hat{d}_i}-\sum_{a=1}^{n}\mathcal{H}(\boldsymbol{r}_i^a,d_i,\kappa,\gamma_i)-\mathrm{i}\hat{d}_i d_i\right)\right\rangle_{\{\gamma_i\}}\\ &=\exp\left(\sum_{i=1}^{V}\log\left\langle\int\prod_{a=1}^{n}\mathrm{d}\boldsymbol{r}_i^a\exp\left(-\sum_{a=1}^{n}\mathcal{H}(\boldsymbol{r}_i^a,d_i,\kappa,\gamma_i)\right)\right.\right.\\ &\left.\left.\times\int_0^{2\pi}\frac{\mathrm{d}\hat{d}_i}{2\pi}\sum_{z=0}^{\infty}\frac{(\mathrm{i}\hat{\rho}(\ldots_i))^z\mathrm{e}^{z\mathrm{i}\hat{d}_i}}{z!}\mathrm{e}^{-\mathrm{i}\hat{d}_i d_i}\right\rangle_{\gamma_i}\right).\end{aligned}$$

Performing the integral over $\hat{d}_i$ to give back a Kronecker delta we see this fixes z equal to d_i and leaves us with

$$\left\langle \int_0^{2\pi} \prod_{i=1}^{V} \frac{\mathrm{d}\hat{d}_i}{2\pi} \prod_{a=1}^{n} \prod_{i=1}^{V} \mathrm{d}\boldsymbol{r}_i^a \prod_{i=1}^{V} \exp\left(\mathrm{i}\hat{\rho}(\ldots_i) \prod_{a=1}^{n} \mathrm{e}^{\mathrm{i}\hat{d}_i} - \sum_{a=1}^{n} \mathcal{H}(\boldsymbol{r}_i^a, d_i, \kappa, \gamma_i) - \mathrm{i}\hat{d}_i d_i \right) \right\rangle_{\{\gamma_i\}}$$
$$= \exp\left(\sum_{i=1}^{V} \log \left\langle \int \prod_{a=1}^{n} \mathrm{d}\boldsymbol{r}_i^a \exp\left(-\sum_{a=1}^{n} \mathcal{H}(\boldsymbol{r}_i^a, d_i, \kappa, \gamma_i) \right) \frac{(\mathrm{i}\hat{\rho}(\ldots_i))^{d_i}}{d_i!} \right\rangle_{\gamma_i} \right).$$

Finally, by introducing the degree distribution $p(d) = \frac{1}{V}\sum_i \delta_{d,d_i}$, we see this may be rewritten as a vertex label independent term given by

$$\left\langle \int_0^{2\pi} \prod_{i=1}^{V} \frac{\mathrm{d}\hat{d}_i}{2\pi} \prod_{a=1}^{n} \prod_{i=1}^{V} \mathrm{d}\boldsymbol{r}_i^a \prod_{i=1}^{V} \exp\left(\mathrm{i}\hat{\rho}(\ldots_i) \prod_{a=1}^{n} \mathrm{e}^{\mathrm{i}\hat{d}_i} - \sum_{a=1}^{n} \mathcal{H}(\boldsymbol{r}_i^a, d_i, \kappa, \gamma_i) - \mathrm{i}\hat{d}_i d_i \right) \right\rangle_{\{\gamma_i\}}$$
$$= \exp\left(V \sum_d p(d) \log \left\langle \int \prod_{a=1}^{n} \mathrm{d}\boldsymbol{r}^a \exp\left(-\sum_{a=1}^{n} \mathcal{H}(\boldsymbol{r}^a, d, \kappa, \gamma) \right) \frac{(\mathrm{i}\hat{\rho}(\ldots))^{d}}{d!} \right\rangle_{\gamma} \right)$$
$$= \exp\left(V S_3[\hat{\rho}] \right). \quad (5.1.18)$$

with

$$S_3[\hat{\rho}] = \sum_d p(d) \log \left\langle \int \prod_{a=1}^{n} \mathrm{d}\boldsymbol{r}^a \exp\left(-\sum_{a=1}^{n} \mathcal{H}(\boldsymbol{r}^a, d, \kappa, \gamma) \right) \frac{\left(\mathrm{i}\hat{\rho}(\boldsymbol{r}^1, \ldots, \boldsymbol{r}^n)\right)^{d}}{d!} \right\rangle_{\gamma}. \quad (5.1.19)$$

Now that the final term of (5.1.15) is rewritten in the form (5.1.18) we have the replicated partition function in an effective single site form given by

$$\langle Z^n \rangle_{\{\gamma_i\},\mathcal{G}} = \frac{1}{\mathcal{N}} \int \mathcal{D}\rho \mathcal{D}\hat{\rho} \exp\left(V \left[\frac{\bar{d}}{2} \left(S_1[\rho] - 1 \right) - \mathrm{i} S_2[\rho, \hat{\rho}] + S_3[\hat{\rho}] \right] \right). \quad (5.1.20)$$

which we can approximate using the saddle point method for large V (see Appendix B.1).

5.1.2 Approximating for large V

For V large, (5.1.20) will be dominated by its saddle point. To proceed however, we must make a specific assumption about the form of the replica densities at the saddle. To derive the earlier cavity equations this will need to be a *replica symmetric assumption* [78].

For the model we are solving symmetric replicas amount to making the ansätze

$$\rho(\boldsymbol{r}^1,\ldots,\boldsymbol{r}^n) = \int \mathcal{D}\psi\,\pi[\psi]\prod_{a=1}^{n}\frac{\exp\left(-\psi(\boldsymbol{r}^a)\right)}{Z[\psi]} \tag{5.1.21}$$

$$\hat{\rho}(\boldsymbol{r}^1,\ldots,\boldsymbol{r}^n) = -\mathrm{i}\bar{d}\int \mathcal{D}\hat{\psi}\,\hat{\pi}[\hat{\psi}]\prod_{a=1}^{n}\frac{\exp\left(-\hat{\psi}(\boldsymbol{r}^a)\right)}{Z[\hat{\psi}]}, \tag{5.1.22}$$

with the convention that

$$Z[f] = \int \mathrm{d}\boldsymbol{r}\exp\left(-f(\boldsymbol{r})\right). \tag{5.1.23}$$

The ansätze (5.1.21) and (5.1.22) assume each single replica function has a Gibbsian form [see e.g. 76]. The average over π and $\hat{\pi}$ can be interpreted as representing, physically, an average[1] over vertices i.

We now substitute the ansätze (5.1.21) and (5.1.22) into (5.1.16) to (5.1.19), and expand in the number of replicas, n, up to $O(n)$. Terms involving λ first feature in the $O(n)$ term of $S_3[\hat{\rho}]$. Since the generalisation error (4.2.2) requires a derivative with respect to λ we must consider both the leading $O(n^0)$ and subleading $O(n)$ terms in calculating our saddle point contributions.

The first two terms in the expansion of $S_1[\rho]$ are calculated as follows: Substituting the ansatz (5.1.21) into (5.1.16) we have

$$S_1[\rho] = \int \mathcal{D}\psi\mathcal{D}\psi'\pi[\psi]\pi[\psi']\left[\int \mathrm{d}\boldsymbol{r}\mathrm{d}(\boldsymbol{r})'\right.$$
$$\left.\frac{\exp\left(-\psi(\boldsymbol{r})-\psi'(\boldsymbol{r}')\right)}{Z[\psi]Z[\psi']}\exp\left(-\mathcal{J}(\boldsymbol{r},(\boldsymbol{r})')\right)\right]^n. \tag{5.1.24}$$

Expanding to order n using $x^n = 1 + n\log x + \ldots$ then gives

$$S_1[\rho] = \int \mathcal{D}\psi\mathcal{D}\psi'\pi[\psi]\pi[\psi']\left[1+\right.$$
$$\left. n\log\int \mathrm{d}\boldsymbol{r}\mathrm{d}\boldsymbol{r}'\frac{\exp\left(-\psi(\boldsymbol{r})-\psi'(\boldsymbol{r}')-\mathcal{J}(\boldsymbol{r},\boldsymbol{r}')\right)}{Z[\psi]Z[\psi']}\right]. \tag{5.1.25}$$

Similarly, applying the same method to $S_2[\rho,\hat{\rho}]$ results in the expansion

$$S_2[\rho,\hat{\rho}] = -i\bar{d}\int \mathcal{D}\psi\mathcal{D}\hat{\psi}\,\pi[\psi]\hat{\pi}[\hat{\psi}]\left[1+n\log\int \mathrm{d}\boldsymbol{r}\frac{\exp\left(-\psi(\boldsymbol{r})-\hat{\psi}(\boldsymbol{r})\right)}{Z[\psi]Z[\hat{\psi}]}\right]. \tag{5.1.26}$$

[1] We have included the prefactor $-\mathrm{i}\bar{d}$ in (5.1.22) because this will ensure, in the end, that both π and $\hat{\pi}$ are normalised probability density functions.

The final term we must expand in powers of n is $S_3(n)[\hat{\rho}]$ as given by (5.1.19). This term however requires a bit more work to write in the desired form: If we substitute the ansatz (5.1.22) into (5.1.19) we have

$$S_3[\hat{\rho}] = \sum_d p(d) \log \left\langle \frac{\bar{d}^d}{d!} \int \prod_{i=1}^{d} \mathcal{D}\hat{\psi}^i \hat{\pi}[\hat{\psi}^i] \left[\int \mathrm{d}\boldsymbol{r} \exp\left(-\mathcal{H}(\boldsymbol{r}, d, \kappa, \gamma)\right) \prod_{i=1}^{d} \frac{\exp\left(-\hat{\psi}^i(\boldsymbol{r})\right)}{Z[\hat{\psi}^i]} \right]^n \right\rangle_\gamma \tag{5.1.27}$$

which, after applying the same $x^n = 1 + n\log(x) + \ldots$ expansion, becomes

$$S_3[\hat{\rho}] = \sum_d p(d) \log \left\langle \frac{\bar{d}^d}{d!} \int \prod_{i=1}^{d} \mathcal{D}\hat{\psi}^i \hat{\pi}[\hat{\psi}^i] \left[1 + n \log \int \mathrm{d}\boldsymbol{r} \exp\left(-\mathcal{H}(\boldsymbol{r}, d, \kappa, \gamma)\right) \prod_{i=1}^{d} \frac{\exp\left(-\hat{\psi}^i(\boldsymbol{r})\right)}{Z[\hat{\psi}^i]} \right] \right\rangle_\gamma . \tag{5.1.28}$$

We see from (5.1.28) that in order to rewrite this as an expansion in n, we must deal with the outer logarithm. We can achieve this by pulling through a factor of $\int \prod_{i=1}^{d} \mathcal{D}\hat{\psi}^i \hat{\pi}[\hat{\psi}^i]$. Performing this step we then see that (5.1.28) becomes

$$S_3[\hat{\rho}] = \sum_d p(d) \left(\log \left[\frac{\bar{d}^d}{d!} \int \prod_{i=1}^{d} \mathcal{D}\hat{\psi}^i \hat{\pi}[\hat{\psi}^i] \right] + \log \left\langle 1 + \frac{n}{\int \prod_{i=1}^{d} \mathcal{D}\hat{\psi}^i \hat{\pi}[\hat{\psi}^i]} \times \int \prod_{i=1}^{d} \mathcal{D}\hat{\psi}^i \hat{\pi}[\hat{\psi}^i] \log \int \mathrm{d}\boldsymbol{r} \exp\left(-\mathcal{H}(\boldsymbol{r}, d, \kappa, \gamma)\right) \prod_{i=1}^{d} \frac{\exp\left(-\hat{\psi}^i(\boldsymbol{r})\right)}{Z[\hat{\psi}^i]} \right\rangle_\gamma \right)$$

which after making the small n approximation $\log(1+nx) \approx nx$ gives the final expanded result

$$S_3[\hat{\rho}] = \sum_d p(d) \left(\log \left[\frac{\bar{d}^d}{d!} \int \prod_{i=1}^{d} \mathcal{D}\hat{\psi}^i \hat{\pi}[\hat{\psi}^i] \right] + \left\langle \frac{n}{\int \prod_{i=1}^{d} \mathcal{D}\hat{\psi}^i \hat{\pi}[\hat{\psi}^i]} \times \int \prod_{i=1}^{d} \mathcal{D}\hat{\psi}^i \hat{\pi}[\hat{\psi}^i] \log \int \mathrm{d}\boldsymbol{r} \exp\left(-\mathcal{H}(\boldsymbol{r}, d, \kappa, \gamma)\right) \prod_{i=1}^{d} \frac{\exp\left(-\hat{\psi}^i(\boldsymbol{r})\right)}{Z[\hat{\psi}^i]} \right\rangle_\gamma \right). \tag{5.1.29}$$

With (5.1.16), (5.1.17) and (5.1.19) in their expanded forms (5.1.25), (5.1.26) and (5.1.29) respectively, we are now in a position to make a saddle point approximation of (5.1.20).

We find that the leading $O(n^0)$ terms cancel with the graph normaliser, $\mathcal{N}$, placing the constraint on π and $\hat{\pi}$ that they must be normalised distributions (see Appendix B for

the normaliser calculation). The subleading saddle point equations together with these $O(n^0)$ normalisation constraints then give a set of simultaneous equations

$$\mu = \bar{d}\int \mathcal{D}\psi'\pi[\psi']\log\int \mathrm{d}\boldsymbol{r}\mathrm{d}\boldsymbol{r}'\frac{\exp\left(-\psi(\boldsymbol{r})-\psi'(\boldsymbol{r}')-\mathcal{J}(\boldsymbol{r},\boldsymbol{r}')\right)}{Z[\psi]Z[\psi']} - \bar{d}\int \mathcal{D}\hat{\psi}\,\hat{\pi}[\hat{\psi}]\log\int \mathrm{d}\boldsymbol{r}\frac{\exp\left(-\psi(\boldsymbol{r})-\hat{\psi}(\boldsymbol{r})\right)}{Z[\psi]Z[\hat{\psi}]} \tag{5.1.30}$$

$$\hat{\mu} = \sum_d p(d)d\int\prod_{i=1}^{d-1}\mathcal{D}\hat{\psi}^i\hat{\pi}[\hat{\psi}^i] \times\left\langle\log\int \mathrm{d}\boldsymbol{r}\frac{\exp\left(-\mathcal{H}(\boldsymbol{r},d,\kappa,\gamma)-\hat{\psi}(\boldsymbol{r})-\sum_{i=1}^{d-1}\hat{\psi}^i(\boldsymbol{r})\right)}{Z[\hat{\psi}]\prod_{i=1}^{d}Z[\hat{\psi}^i]}\right\rangle_\gamma - \bar{d}\int \mathcal{D}\psi\,\pi[\psi]\log\int \mathrm{d}\boldsymbol{r}\frac{\exp\left(-\psi(\boldsymbol{r})-\hat{\psi}(\boldsymbol{r})\right)}{Z[\psi]Z[\hat{\psi}]} \tag{5.1.31}$$

which must be satisfied for π and $\hat{\pi}$ to be at a saddle point and where the $O(n^0)$ normalisation constraints have been enforced with Lagrange multipliers μ and $\hat{\mu}$. Looking at (5.1.30), we see the $Z[\psi']$ and $Z[\hat{\psi}]$ factors give additive constants that can be absorbed, along with the factors of $\bar{d}$, into a redefinition of μ. Defining

$$\hat{\Psi}[\psi'](\boldsymbol{r}) = -\log\int \mathrm{d}\boldsymbol{r}'\exp\left(-\psi'(\boldsymbol{r}')-\mathcal{J}(\boldsymbol{r},\boldsymbol{r}')\right). \tag{5.1.32}$$

we see the resulting equation has the form

$$\mu = \int \mathcal{D}\psi'\pi[\psi']F(\psi,\hat{\Psi}[\psi']) - \int \mathcal{D}\hat{\psi}\,\hat{\pi}[\hat{\psi}]F(\psi,\hat{\psi}) \tag{5.1.33}$$

with $F(\psi,\hat{\psi}) = \log\int \mathrm{d}\boldsymbol{r}\,Z^{-1}[\psi]\exp\left(-\psi(\boldsymbol{r})-\hat{\psi}(\boldsymbol{r})\right)$. As this equation must hold for arbitrary ψ, it follows that the distribution of $\hat{\psi}$ must equal the distribution of $\hat{\Psi}[\psi']$, hence[2]

$$\hat{\pi}[\hat{\psi}] = \int \mathcal{D}\psi\pi[\psi]\delta(\hat{\psi}-\hat{\Psi}[\psi]). \tag{5.1.34}$$

Similarly one derives from (5.1.31) that

$$\pi[\psi] = \sum_d \frac{p(d)d}{\bar{d}}\left\langle\int\prod_{i=1}^{d-1}\mathcal{D}\hat{\psi}^i\hat{\pi}[\hat{\psi}^i]\delta(\psi-\Psi[\hat{\psi}^1,\ldots,\hat{\psi}^{d-1}])\right\rangle_\gamma, \tag{5.1.35}$$

[2]To be precise, $\hat{\psi}$ and $\hat{\Psi}[\psi']$ can differ by an arbitrary additive constant because $F(\psi,\hat{\psi}+\mathrm{const}) = F(\psi,\hat{\psi})+\mathrm{const}$; we fix this constant to the most convenient value in writing (5.1.34).

with Ψ defined as

$$\Psi[\hat{\psi}^1,\dots,\hat{\psi}^{d-1}](\boldsymbol{r}) = \sum_{i=1}^{d-1}\hat{\psi}^i(\boldsymbol{r}) + \mathcal{H}(\boldsymbol{r},d,\kappa,\gamma). \tag{5.1.36}$$

Combining (5.1.34) and (5.1.35) we finally arrive at the self-consistency equation

$$\pi[\psi] = \sum_d \frac{p(d)d}{\bar{d}} \int \prod_{i=1}^{d-1} \mathcal{D}\psi^i \pi[\psi^i] \left\langle \delta(\psi - \Psi[\hat{\Psi}[\psi^1],\dots,\hat{\Psi}[\psi^{d-1}]]) \right\rangle_\gamma \tag{5.1.37}$$

which is the analogue of the cavity update equations given by (4.2.8).

Looking at the explicit form of $\mathcal{J}$ and $\mathcal{H}$, similar to the cavity derivation, one sees that these equations are solved by 'energy functions' $\psi(\boldsymbol{r})$ that contain only quadratic terms in $\boldsymbol{r}$. We write these functions in terms of their (complex valued) covariance matrices $\boldsymbol{V}$ so that $\psi(\boldsymbol{r}) = \frac{1}{2}\boldsymbol{r}^{\mathrm{T}}\boldsymbol{V}^{-1}\boldsymbol{r}$. In terms of the distribution of the covariance matrices, $\boldsymbol{V}$, the self-consistency equation (5.1.37) becomes

$$\pi[\boldsymbol{V}] = \sum_d \frac{p(d)d}{\bar{d}} \int \prod_{i=1}^{d-1} \mathrm{d}\boldsymbol{V}^i \pi[\boldsymbol{V}^i] \left\langle \delta\Big(\boldsymbol{V} - \Big[\boldsymbol{O} - \sum_{i=1}^{d-1} \boldsymbol{X}\boldsymbol{V}^i\boldsymbol{X}\Big]^{-1}\Big) \right\rangle_\gamma, \tag{5.1.38}$$

where $\boldsymbol{O}$ and $\boldsymbol{X}$ are defined in (4.2.10) and (4.2.11) respectively. One can see, as expected, that (5.1.38) is exactly the same equation as (4.2.13) derived in Chapter 4.

Once more, as was the case for the cavity derivation, this equation seems to have a singularity when γ equals zero which can be resolved using the Woodbury identity (see (4.2.15)). Once we have the distribution π the generalisation error is calculated using (see Appendix B)

$$\epsilon_g(\nu) = -2\lim_{\lambda\to 0}\frac{\partial S_3^{O(n)}}{\partial\lambda} = \left\langle \sum_d p(d) \int \prod_{k=1}^{d} \mathrm{d}\boldsymbol{V}_k\, \pi[\boldsymbol{V}_k]\, \frac{1}{\gamma/\sigma^2 + d\kappa \boldsymbol{e}_0^{\mathrm{T}}\boldsymbol{M}_d^{-1}\boldsymbol{e}_0} \right\rangle_\gamma, \tag{5.1.39}$$

where we have defined $S_3^{O(n)}$ as the $O(n)$ terms in the small-n expansion of S_3 (see (5.1.19)), used the definition just after (4.2.11) for $\boldsymbol{M}_d$, defined $\boldsymbol{e}_0 = (1,0,\dots,0)^{\mathrm{T}}$ and again taken advantage of the Woodbury identity [48]. Similarly to (5.1.37) we see that the generalisation error is again the same result that we derived using cavities in Chapter 4 (see (4.2.16)).

5.2 Local normalisation

Using the insight gained from the simpler globally normalised kernel, we will now derive a prediction of the generalisation error for the more complicated case of a locally normalised kernel. Here κ_i is equal to the prior variance at vertex i, as calculated from the unnormalised kernel. This change in normalisation requires that properties of the quenched disorder are effectively defined via a second thermal average, making the calculation rather more complicated. The replica treatment of this calculation will give a deeper insight into the cavity update equation (4.2.26) and the approximations required to derive it.

We begin our calculation for the locally normalised kernel by including the vertex-dependent normalisation constants κ_i in (5.1.3) via a delta enforcement term

$$Z = \int \prod_{i=1}^{V} \mathrm{d}\boldsymbol{r}_i \mathrm{d}\boldsymbol{\kappa} \prod_{i=1}^{V} \exp\left(-\mathcal{H}(\boldsymbol{r}_i, d_i, \kappa_i, \gamma_i)\right) \prod_{i<j} \exp\left(-a_{ij}\mathcal{J}(\boldsymbol{r}_i, \boldsymbol{r}_j)\right) \times \prod_{i=1}^{V} \delta\left(\kappa_i - \frac{1}{Z_{\mathrm{aux}}} \int \mathrm{d}\boldsymbol{f}_{\mathrm{aux}} (f_{\mathrm{aux},i})^2 \exp\left(-\frac{1}{2}\boldsymbol{f}_{\mathrm{aux}}{}^{\mathrm{T}} \hat{\boldsymbol{C}}^{-1} \boldsymbol{f}_{\mathrm{aux}}\right)\right). \quad (5.2.1)$$

Here Z_{aux} is defined to be the normaliser for an auxiliary GP $\boldsymbol{f}_{\mathrm{aux}}$, $\hat{\boldsymbol{C}} = (\boldsymbol{I} - a\boldsymbol{L})^p$ is the unnormalised kernel from (4.1.1) and $\boldsymbol{\kappa} = (\kappa_1, \ldots, \kappa_V)^{\mathrm{T}}$.

Because the definitions of κ_i depend on the entire graph in a non-local way, the partition function (5.2.1) no longer has the structure of a graphical model. Before we can apply the replica method used in section 5.1 therefore, we need to bring it back into this form. This will require the introduction of an additional set of replicas for the auxiliary GP.

Focussing on the κ_i-enforcement terms, we rewrite the delta functions in terms of their Fourier transform by introducing conjugate variables $\hat{\kappa}_i$, and replacing the integral over $\boldsymbol{f}_{\mathrm{aux}}$ in the exponent with an empirical average over L replicas of $\boldsymbol{f}_{\mathrm{aux}}$. We then let L tend to infinity later to retrieve the true average in (5.2.1). With an element of foresight we also introduce a small regularising term λ_{aux} so that later equations are well behaved for $\hat{\kappa}_i$ equal to zero; λ_{aux} will then be set equal to zero at the end. With the introduction of the L replicas and regulariser λ_{aux}, the delta enforcement terms from (5.2.1) are given

by

$$\lim_{\substack{L\to\infty\\ \lambda_{\rm aux}\to 0}} \int \prod_{i=1}^{V} \frac{\mathrm{d}\hat{\kappa}_i}{2\pi} \prod_{l=1}^{L} \left(\frac{\mathrm{d}\boldsymbol{f}^l_{\rm aux}}{Z_{\rm aux}} \right) \\ \times \exp\left(\sum_{i=1}^{V} \left[\mathrm{i}\hat{\kappa}_i \kappa_i - \frac{1}{2} \left(\frac{2\mathrm{i}\hat{\kappa}_i}{L} + \lambda_{\rm aux} \right) \sum_{l=1}^{L} (f^l_{{\rm aux},i})^2 \right] - \frac{1}{2} \sum_{l=1}^{L} (\boldsymbol{f}^l_{\rm aux})^{\mathrm{T}} \hat{\boldsymbol{C}}^{-1} \boldsymbol{f}^l_{\rm aux} \right). \quad (5.2.2)$$

The non-local coupling via $\hat{\boldsymbol{C}}^{-1}$ can be reduced to nearest neighbour interactions by the same method we deployed in deriving (4.2.6) from (4.2.3) for the original GP variables $\boldsymbol{f}$. In order to get (5.2.1) into the form of a graphical model it then remains to deal with the graph dependent $Z_{\rm aux}$ term. We bring $Z_{\rm aux}$ into the numerator by introducing $m-1$ replicas of $Z_{\rm aux}$ and taking m to zero at the end so that $Z_{\rm aux}^{-1} = \lim_{m\to 0} Z_{\rm aux}^{m-1}$. As is conventional in replica calculations, we then exchange the limits m to zero and L to infinity, taking the latter first. The L-replica average over $(f^l_{{\rm aux},i})^2 \equiv (f^{l,1}_{{\rm aux},i})^2$ can next be symmetrised to an Lm-replica average over the $(f^{l,b}_{{\rm aux},i})^2$ with $b = 1, \ldots, m$. One can see that this is possible by realising that we have an infinite number mL of replicas at any fixed $m > 0$ so that both averages become non-fluctuating. At this point the only effect of m is via the total number of replicas mL, and to simplify the notation we can fix $m = 1$. At the end of the calculation of $\langle \log Z \rangle$ we then in principle have to replace L by mL and take m to zero. However, since for the generalisation error we only need the λ-derivative, which does not directly depend on the auxiliary degrees of freedom, this will not be necessary.

With these simplifications, and after also rescaling $\hat{\kappa}_i$ by a factor of L for later convenience, the delta function constraints for the κ_i take the form

$$\lim_{\substack{\lambda_{\rm aux}\to 0\\ L\to\infty}} \int \prod_{i=1}^{V} \frac{\mathrm{d}\hat{\kappa}_i L}{2\pi} \prod_{l=1}^{L} \mathrm{d}\boldsymbol{f}^l_{\rm aux} \exp\left(\sum_{i=1}^{V} \left[\mathrm{i}L\hat{\kappa}_i \kappa_i - \frac{1}{2} \left(2\mathrm{i}\hat{\kappa}_i + \lambda_{\rm aux} \right) \sum_{l=1}^{L} (f^l_{{\rm aux},i})^2 \right] \right) \\ \times \exp\left(-\frac{1}{2} \sum_{l=1}^{L} (\boldsymbol{f}^l_{\rm aux})^{\mathrm{T}} \hat{\boldsymbol{C}}^{-1} \boldsymbol{f}^l_{\rm aux} \right). \quad (5.2.3)$$

Finally rewriting (5.2.3) using the same techniques as we did for Z in section 4.2.1, by Fourier transforming the covariance, integrating out the remaining terms, introducing $2p$ additional variables at each vertex, rescaling according to $(h^*)^q_{{\rm aux},i} = h^q_{{\rm aux},i} d_i^{-1/2}$ and $(\hat{h}^*)^q_{{\rm aux},i} = \hat{h}^q_{{\rm aux},i} d_i^{-1/2}$ and dropping the asterisks again, we arrive at the desired

graphical model form of the partition function:

$$Z = \lim_{\substack{\lambda_{\text{aux}}\to 0\\ L\to\infty}} \int \prod_{i=1}^{V} \mathrm{d}\boldsymbol{r}_i \, \frac{\mathrm{d}\boldsymbol{\kappa}\mathrm{d}\hat{\boldsymbol{\kappa}} L^V}{(2\pi)^V} \prod_{l=1}^{L}\prod_{i=1}^{V} \mathrm{d}\boldsymbol{r}^l_{\text{aux},i} \times \prod_{i=1}^{V} \exp\left(-\mathcal{H}(\boldsymbol{r}_i, d_i, \kappa_i, \gamma_i) + \mathrm{i}L\kappa_i\hat{\kappa}_i - \sum_{l=1}^{L} \mathcal{H}_{\text{aux}}(\boldsymbol{r}^l_{\text{aux},i}, d_i, \hat{\kappa}_i) \right) \times \prod_{i<j} \exp\left(-a_{ij}\mathcal{J}(\boldsymbol{r}_i, \boldsymbol{r}_j) - \sum_{l=1}^{L} a_{ij}\mathcal{J}(\boldsymbol{r}^l_{\text{aux},i}, \boldsymbol{r}^l_{\text{aux},j}) \right). \quad (5.2.4)$$

where we have defined

$$\mathcal{H}_{\text{aux}}(\boldsymbol{r}_{\text{aux}}, d, \hat{\kappa}) = \frac{d}{2}\sum_{q=0}^{p} c_q h^0_{\text{aux}} h^q_{\text{aux}} + \frac{d}{2}\frac{(h^0_{\text{aux}})^2}{\lambda_{\text{aux}} + 2\mathrm{i}\hat{\kappa}} - \mathrm{i}d\sum_{q=1}^{p} \hat{h}^q_{\text{aux}} h^q_{\text{aux}} - \frac{1}{2}\log\left(\frac{1}{\lambda_{\text{aux}} + 2\mathrm{i}\hat{\kappa}}\right). \quad (5.2.5)$$

5.2.1 Deriving the 'effective single site' formulation

With (5.2.1) now in a form suitable for the application of the replica method used in section 5.1 we proceed initially as we did in the case of global normalisation in section 5.1.1; we bring the graph and input average inside the logarithm using the replica trick at the cost of n additional replicas. The resulting average of Z^n reads as,

$$\langle Z^n \rangle_{\{\gamma_i\},\mathcal{G}} = \lim_{L\to\infty} \left\langle \int \prod_{a=1}^{n} \left(\prod_{i=1}^{V} \mathrm{d}\boldsymbol{r}^a_i \, \frac{\mathrm{d}\boldsymbol{\kappa}^a \mathrm{d}\hat{\boldsymbol{\kappa}}^a L^V}{(2\pi)^V} \prod_{l=1}^{L}\prod_{i=1}^{V} \mathrm{d}\boldsymbol{r}^{l,a}_{\text{aux},i} \right) \times \prod_{i<j} \exp\left(-\sum_{a=1}^{n} a_{ij}\mathcal{J}(\boldsymbol{r}^a_i, \boldsymbol{r}^a_j) - \sum_{a=1}^{n}\sum_{l=1}^{L} a_{ij}\mathcal{J}(\boldsymbol{r}^{l,a}_{\text{aux},i}, \boldsymbol{r}^{l,a}_{\text{aux},j}) \right) \times \prod_{i=1}^{V} \exp\left(-\sum_{a=1}^{n} \mathcal{H}(\boldsymbol{r}^a_i, d_i, \kappa_i, \gamma_i) + \mathrm{i}L\sum_{a=1}^{n} \kappa^a_i \hat{\kappa}^a_i - \sum_{a=1}^{n}\sum_{l=1}^{L} \mathcal{H}_{\text{aux}}(\boldsymbol{r}^{l,a}_{\text{aux},i}, d_i, \hat{\kappa}^a_i) \right) \right\rangle_{\{\gamma_i\},\mathcal{G}} \quad (5.2.6)$$

Similarly to the global case we introduce replica densities

$$\rho(\boldsymbol{r}^1, \ldots, \boldsymbol{r}^n, \boldsymbol{r}^{1,1}_{\text{aux}}, \ldots, \boldsymbol{r}^{n,L}_{\text{aux}}, \kappa^1, \ldots, \kappa^n, \hat{\kappa}^1, \ldots, \hat{\kappa}^n) = \frac{1}{V}\sum_i \mathrm{e}^{\mathrm{i}\hat{d}_i} \prod_{a=1}^{n} \delta(\boldsymbol{r}^a - \boldsymbol{r}^a_i)\delta(\kappa^a - \kappa^a_i)\delta(\hat{\kappa}^a - \hat{\kappa}^a_i) \prod_{l=1}^{L} \delta(\boldsymbol{r}^{l,a}_{\text{aux}} - \boldsymbol{r}^{l,a}_{\text{aux},i}) \quad (5.2.7)$$

with enforcement via conjugate densities $\hat{\rho}$. We substitute the densities into (5.2.6) and perform the average over the graph ensemble to derive, using techniques similar to those for the global case (see equations (5.1.11) through to (5.1.20)), an effective single site formulation

$$\langle Z^n \rangle_{\{\gamma_i\},\mathcal{G}} = \frac{1}{\mathcal{N}} \int \mathcal{D}\rho \mathcal{D}\hat{\rho} \exp\left(V \left[\frac{\bar{d}}{2}(S_1[\rho] - 1) - \mathrm{i}S_2[\rho, \hat{\rho}] + S_3[\hat{\rho}] \right] \right), \tag{5.2.8}$$

where

$$S_1[\rho] = \int \prod_{a=1}^{n} \left(\mathrm{d}\boldsymbol{r}^a \mathrm{d}(\boldsymbol{r}^a)' \frac{\mathrm{d}\kappa^a \mathrm{d}\hat{\kappa}^a L}{2\pi} \frac{\mathrm{d}(\kappa^a)' \mathrm{d}(\hat{\kappa}^a)' L}{2\pi} \prod_{l=1}^{L} \mathrm{d}\boldsymbol{r}^{l,a}_{\mathrm{aux}} \mathrm{d}(\boldsymbol{r}^{l,a}_{\mathrm{aux}})' \right) \rho(\ldots)\rho(\ldots') \\ \times \exp\left(-\sum_{a=1}^{n} \mathcal{J}(\boldsymbol{r}^a, (\boldsymbol{r}^a)') - \sum_{a=1}^{n} \sum_{l=1}^{L} \mathcal{J}(\boldsymbol{r}^{l,a}_{\mathrm{aux}}, (\boldsymbol{r}^{l,a}_{\mathrm{aux}})') \right) \tag{5.2.9}$$

$$S_2[\rho, \hat{\rho}] = \int \prod_{a=1}^{n} \left(\mathrm{d}\boldsymbol{r}^a \frac{\mathrm{d}\kappa^a \mathrm{d}\hat{\kappa}^a L}{2\pi} \prod_{l=1}^{L} \mathrm{d}\boldsymbol{r}^{l,a}_{\mathrm{aux}} \right) \rho(\ldots)\hat{\rho}(\ldots) \tag{5.2.10}$$

$$S_3[\hat{\rho}] = \sum_d p(d) \log \left\langle \int \prod_{a=1}^{n} \left(\mathrm{d}\boldsymbol{r}^a \frac{\mathrm{d}\kappa^a \mathrm{d}\hat{\kappa}^a L}{2\pi} \prod_{l=1}^{L} \mathrm{d}\boldsymbol{r}^{l,a}_{\mathrm{aux}} \right) \exp\left(-\sum_{a=1}^{n} \mathcal{H}(\boldsymbol{r}^a, d, \kappa^a, \gamma) \right. \right. \\ \left. \left. + \mathrm{i}L \sum_{a=1}^{n} \kappa^a \hat{\kappa}^a - \sum_{a=1}^{n} \sum_{l=1}^{L} \mathcal{H}_{\mathrm{aux}}(\boldsymbol{r}^{l,a}_{\mathrm{aux}}, d, \hat{\kappa}^a) \right) \frac{(\mathrm{i}\hat{\rho}(\ldots))^d}{d!} \right\rangle_{\gamma}. \tag{5.2.11}$$

Note, that in order to save space the limits λ_{aux} to zero and L infinity have been omitted above.

5.2.2 Approximating for large V

As before, the expression (5.2.8) for the average of the replicated partition function will be dominated by its saddle point for large V. To tie in with the cavity derivation we once more assume that this saddle point has a replica symmetric form and set

$$\rho(\ldots) = \int \mathcal{D}\psi \, \pi[\psi] \prod_{a=1}^{n} \frac{\exp\left(-\psi(\boldsymbol{r}^a, \kappa^a, \hat{\kappa}^a, \{\boldsymbol{r}^{l,a}_{\mathrm{aux}}\}) \right)}{Z[\psi]} \tag{5.2.12}$$

$$\hat{\rho}(\ldots) = -\mathrm{i}\bar{d} \int \mathcal{D}\hat{\psi} \, \hat{\pi}[\hat{\psi}] \prod_{a=1}^{n} \frac{\exp\left(-\hat{\psi}(\boldsymbol{r}^a, \kappa^a, \hat{\kappa}^a, \{\boldsymbol{r}^{l,a}_{\mathrm{aux}}\}) \right)}{Z[\hat{\psi}]}. \tag{5.2.13}$$

Substituting (5.2.12) and (5.2.13) into (5.2.9) to (5.2.11), we can write each exponent function as an expansion in n, setting $S_i = S_i^{O(1)} + nS_i^{O(n)}$ up to linear order in n. As

before, and as is standard in replica calculations, the subleading $O(n)$ terms are required in order to calculate the generalisation error. The novel feature of this calculation is the dependence on L replicas of the auxiliary variables. We will see that this leads to equations that couple $\boldsymbol{r}$ to $\{\boldsymbol{r}^l_{\text{aux}}\}$ via the covariance matrix of the latter.

With the replica densities introduced one finds that the expansions for S_1, S_2 and S_3 can be calculated in a similar manner to the global case (see equations (5.1.24) through to (5.1.29)). Leading $O(n^0)$ terms once more cancel the normaliser, $\mathcal{N}$, at the saddle point (see Appendix B), and we are left with optimising $O(n)$ terms given by

$$S_1^{O(n)}[\rho] = \int \mathcal{D}\psi\, \pi[\psi] \int \mathcal{D}\psi'\, \pi[\psi'] \log \Bigg[\int \mathrm{d}\boldsymbol{r} \frac{\mathrm{d}\kappa \mathrm{d}\hat{\kappa} L}{2\pi} \mathrm{d}\boldsymbol{r}' \frac{\mathrm{d}\kappa' \mathrm{d}\hat{\kappa}' L}{2\pi} \prod_l \mathrm{d}\boldsymbol{r}^l_{\text{aux}} \mathrm{d}(\boldsymbol{r}^l_{\text{aux}})' \exp\Big(-\psi(\boldsymbol{r}, \kappa, \hat{\kappa}, \{\boldsymbol{r}^l_{\text{aux}}\}) - \psi'(\boldsymbol{r}', \kappa', \hat{\kappa}', \{(\boldsymbol{r}^l_{\text{aux}})'\})\Big) \times \frac{\exp\left(-\mathcal{J}(\boldsymbol{r}, \boldsymbol{r}') - \sum_l \mathcal{J}(\boldsymbol{r}^l_{\text{aux}}, (\boldsymbol{r}^l_{\text{aux}})')\right)}{Z[\psi] Z[\psi']} \Bigg] \tag{5.2.14}$$

$$S_2^{O(n)}[\rho, \hat{\rho}] = -\mathrm{i}\bar{d} \int \mathcal{D}\psi\, \pi[\psi] \int \mathcal{D}\hat{\psi}\, \hat{\pi}[\hat{\psi}] \log \Bigg[\int \mathrm{d}\boldsymbol{r} \frac{\mathrm{d}\kappa \mathrm{d}\hat{\kappa} L}{2\pi} \prod_l \mathrm{d}\boldsymbol{r}^l_{\text{aux}} \frac{\exp\Big(-\psi(\boldsymbol{r}, \kappa, \hat{\kappa}, \{\boldsymbol{r}^l_{\text{aux}}\}) - \hat{\psi}(\boldsymbol{r}, \kappa, \hat{\kappa}, \{\boldsymbol{r}^l_{\text{aux}}\})\Big)}{Z[\psi] Z[\hat{\psi}]} \Bigg] \tag{5.2.15}$$

$$S_3^{O(n)}[\hat{\rho}] = \sum_d p(d) \Bigg\langle \int \prod_{i=1}^{d} \mathcal{D}\hat{\psi}^i\, \hat{\pi}[\hat{\psi}^i] \log \Bigg[\int \mathrm{d}\boldsymbol{r} \frac{\mathrm{d}\kappa \mathrm{d}\hat{\kappa} L}{2\pi} \prod_l \mathrm{d}\boldsymbol{r}^l_{\text{aux}} \exp\left(-\mathcal{H}(\boldsymbol{r}, d, \kappa, \gamma) + iL\kappa\hat{\kappa} - \sum_l \mathcal{H}_{\text{aux}}(\boldsymbol{r}^l_{\text{aux}}, d, \hat{\kappa})\right) \times \frac{\exp\left(-\sum_{i=1}^{d} \hat{\psi}^i(\boldsymbol{r}, \kappa, \hat{\kappa}, \{\boldsymbol{r}^l_{\text{aux}}\})\right)}{\prod_{i=1}^{d} Z[\hat{\psi}^i]} \Bigg] \Bigg\rangle_{\gamma}, \tag{5.2.16}$$

subject to π and $\hat{\pi}$ being normalised density functions.

Proceeding as we did in the global case (see (5.1.30) through to (5.1.37)) we see that the saddle point conditions with respect to π and $\hat{\pi}$ can be calculated as

$$\pi[\psi] = \sum_d \frac{p(d) d}{\bar{d}} \int \prod_{i=1}^{d-1} \mathcal{D}\psi^i\, \pi[\psi^i] \Big\langle \delta(\psi - \Psi[\hat{\Psi}[\psi^1], \ldots, \hat{\Psi}[\psi^{d-1}]]) \Big\rangle_{\gamma}, \tag{5.2.17}$$

with

$$\Psi[\hat{\psi}^1,\ldots,\hat{\psi}^{d-1}] = \sum_{i=1}^{d-1}\hat{\psi}^i(\boldsymbol{r},\kappa,\hat{\kappa},\{\boldsymbol{r}^l_{\text{aux}}\}) - \mathrm{i}L\kappa\hat{\kappa} + \mathcal{H}(\boldsymbol{r},d,\kappa,\gamma) + \sum_l \mathcal{H}_{\text{aux}}(\boldsymbol{r}^l_{\text{aux}},d,\hat{\kappa}), \quad (5.2.18)$$

$$\hat{\Psi}[\psi'] = -\log\int \mathrm{d}\boldsymbol{r}'\frac{\mathrm{d}\kappa'\mathrm{d}\hat{\kappa}'L}{2\pi}\prod_l \mathrm{d}(\boldsymbol{r}^l_{\text{aux}})'\exp\left(-\psi'(\boldsymbol{r}',\kappa',\hat{\kappa}',\{(\boldsymbol{r}^l_{\text{aux}})'\})\right) \times\exp\left(-\mathcal{J}(\boldsymbol{r},\boldsymbol{r}') - \prod_l \mathcal{J}(\boldsymbol{r}^l_{\text{aux}},(\boldsymbol{r}^l_{\text{aux}})')\right). \quad (5.2.19)$$

Equations (5.2.17), (5.2.18) and (5.2.19) are just the analogues of the saddle point conditions for the globally normalised kernel ((5.1.37), (5.1.36) and (5.1.32) respectively), for a larger set of variables. Similarly an analogous expression for the generalisation error (see (5.1.39)) can be calculated as,

$$\epsilon_g = \lim_{\lambda\to 0}\sum_d p(d)\left\langle\int\prod_{i=1}^d \mathcal{D}\psi^i\pi[\psi^i]\int \mathrm{d}\boldsymbol{r}\frac{\mathrm{d}\kappa\mathrm{d}\hat{\kappa}L}{2\pi}\prod_{l=1}^L \mathrm{d}\boldsymbol{r}^l_{\text{aux}} \left(\frac{1}{\gamma/\sigma^2+\lambda} - \frac{\kappa d\boldsymbol{e}_0^{\mathrm{T}}\boldsymbol{r}\boldsymbol{r}^{\mathrm{T}}\boldsymbol{e}_0}{(\gamma/\sigma^2+\lambda)^2}\right) q(\boldsymbol{r},\kappa,\hat{\kappa},\{\boldsymbol{r}^l_{\text{aux}}\})\right\rangle_\gamma, \quad (5.2.20)$$

with

$$q(\boldsymbol{r},\kappa,\hat{\kappa},\{\boldsymbol{r}^l_{\text{aux}}\}) \propto \exp\left(-\Psi[\hat{\Psi}[\psi^1],\ldots,\hat{\Psi}[\psi^d]](\boldsymbol{r},\kappa,\hat{\kappa},\{\boldsymbol{r}^l_{\text{aux}}\})\right), \quad (5.2.21)$$

and $\boldsymbol{e}_0$ defined as before as $\boldsymbol{e}_0 = (1,0,\ldots,0)^{\mathrm{T}}$. The proportionality factor in (5.2.21) is determined so that q is normalised with respect to integration over $\boldsymbol{r}$, $\boldsymbol{r}_{\text{aux}}$, κ and $\hat{\kappa}L/(2\pi)$.

5.2.3 Approximating for large L

While the replica calculation for local normalisation has so far broadly followed that of the global case in section 5.1, we now need to take some additional steps to deal with the fact that the number of auxiliary variables diverges in the desired L large limit. Fortunately this limit allows us to make simplifications in both the saddle point condition (5.2.17) and the expression (5.2.20) of the generalisation error. By using large L-saddle point methods we will obtain a numerically tractable set of equations from which we will be able to derive the generalisation error approximation given in (4.2.28).

Closer inspection of (5.2.19) shows that $\hat{\Psi}[\psi]$ only depends on the marginal

$$\phi(\boldsymbol{r}, \{\boldsymbol{r}^l_{\text{aux}}\}) = -\log \int \mathrm{d}\kappa \mathrm{d}\hat{\kappa} L/(2\pi) \exp(-\psi(\boldsymbol{r}, \kappa, \hat{\kappa}, \{\boldsymbol{r}^l_{\text{aux}}\})), \tag{5.2.22}$$

and so $\hat{\Psi}[\psi]$ is also itself independent of κ and $\hat{\kappa}$. To emphasise this we will write the function it produces as $\hat{\phi}$ and the functional itself as $\hat{\Phi}[\phi]$.

With this realisation we see that in the limit of large L, the relevant marginalised (over κ and $\hat{\kappa}$) versions of (5.2.18) and (5.2.19) can now be calculated by a saddle point evaluation (see Appendix B.1.2). Explicitly, we see that the marginal of the functional $\Psi[\hat{\psi}^1, \ldots, \hat{\psi}^{d-1}]$ in (5.2.18) over κ and $\hat{\kappa}$ is given by

$$\Phi[\hat{\phi}^1, \ldots, \hat{\phi}^{d-1}] = -\log \int \frac{\mathrm{d}\kappa \mathrm{d}\hat{\kappa} L}{2\pi} \zeta(\boldsymbol{r}, \kappa, \hat{\kappa}, \{\boldsymbol{r}^l_{\text{aux}}\}) \exp\left(-L\xi(\{\boldsymbol{r}^l_{\text{aux}}\}, \kappa, \hat{\kappa})\right), \tag{5.2.23}$$

with

$$\zeta(\boldsymbol{r}, \kappa, \hat{\kappa}, \{\boldsymbol{r}^l_{\text{aux}}\}) = \exp\left(-\mathcal{H}(\boldsymbol{r}, d, \kappa, \gamma) - \sum_{i=1}^{d-1} \hat{\phi}^i(\boldsymbol{r}, \{\boldsymbol{r}^l_{\text{aux}}\})\right) \tag{5.2.24}$$

$$\xi(\{\boldsymbol{r}^l_{\text{aux}}\}, \kappa, \hat{\kappa}) = \frac{1}{L} \sum_l \mathcal{H}_{\text{aux}}(\boldsymbol{r}^l_{\text{aux}}, d, \hat{\kappa}) - \mathrm{i}\kappa\hat{\kappa}. \tag{5.2.25}$$

and so the saddle point conditions of (5.2.23) are

$$0 = -\mathrm{i}\hat{\kappa} \tag{5.2.26}$$

$$0 = \frac{\mathrm{i}}{\lambda_{\text{aux}} + 2\mathrm{i}\hat{\kappa}} - \mathrm{i}\kappa - \mathrm{i}\frac{d}{L} \sum_l \frac{\boldsymbol{e}_0^{\mathrm{T}} \boldsymbol{r}^l_{\text{aux}} (\boldsymbol{r}^l_{\text{aux}})^{\mathrm{T}} \boldsymbol{e}_0}{(\lambda_{\text{aux}} + 2\mathrm{i}\hat{\kappa})^2}. \tag{5.2.27}$$

This gives us the saddle point values

$$\kappa = \frac{1}{\lambda_{\text{aux}}} - \frac{d}{L} \sum_l \frac{\boldsymbol{e}_0^{\mathrm{T}} \boldsymbol{r}^l_{\text{aux}} (\boldsymbol{r}^l_{\text{aux}})^{\mathrm{T}} \boldsymbol{e}_0}{\lambda^2_{\text{aux}}} \tag{5.2.28}$$

$$\hat{\kappa} = 0. \tag{5.2.29}$$

Unlike the earlier saddle point approximation in section 5.1 where prefactors cancel, since this result must be substituted into (5.2.23) we must also calculate the $O(1)$ prefactors. To obtain these prefactors one needs to expand the exponent of (5.2.23) to second order about the saddle point and ensure we pass through the saddle point along a plane with constant imaginary exponent (see Appendix B.1.2). Fortunately the relevant Hessian of $\xi(\{\boldsymbol{r}^l_{\text{aux}}\}, \kappa, \hat{\kappa})$ is given by

$$\begin{pmatrix} 0 & -\mathrm{i} \\ -\mathrm{i} & \frac{2}{(\lambda_{\text{aux}} + 2\mathrm{i}\hat{\kappa})^2} - \frac{4d}{L} \sum_l \frac{\boldsymbol{e}_0^{\mathrm{T}} \boldsymbol{r}^l_{\text{aux}} (\boldsymbol{r}^l_{\text{aux}})^{\mathrm{T}} \boldsymbol{e}_0}{(\lambda_{\text{aux}} + 2\mathrm{i}\hat{\kappa})^3} \end{pmatrix} \tag{5.2.30}$$

which has a constant determinant and tells us that we must approach the saddle along the real line for both κ and $\hat{\kappa}$. The factor from the fluctuation correction therefore is simply $(2\pi/L)$. This just cancels the scaling factor in front of the integral over κ and $\hat{\kappa}$. Substituting the saddle point approximation into $\Phi[\hat{\phi}^1, \ldots, \hat{\phi}^{d-1}]$ gives

$$\Phi[\hat{\phi}^1, \ldots, \hat{\phi}^{d-1}](\boldsymbol{r}, \{\boldsymbol{r}^l_{\text{aux}}\}) = \sum_{i=1}^{d-1} \hat{\phi}^i(\boldsymbol{r}, \{\boldsymbol{r}^l_{\text{aux}}\}) + \mathcal{H}\Big(\boldsymbol{r}, d, v(\boldsymbol{H}), \gamma\Big) + \sum_l \mathcal{H}_{\text{aux}}(\boldsymbol{r}^l_{\text{aux}}, d, 0), \quad (5.2.31)$$

$$\hat{\Phi}[\phi'] = -\log \int \mathrm{d}\boldsymbol{r}' \prod_l \mathrm{d}(\boldsymbol{r}^l_{\text{aux}})' \exp\left(-\phi'(\boldsymbol{r}', \{(\boldsymbol{r}^l_{\text{aux}})'\}) - \mathcal{J}(\boldsymbol{r}, \boldsymbol{r}') \right) \times \exp\left(-\prod_l \mathcal{J}(\boldsymbol{r}^l_{\text{aux}}, (\boldsymbol{r}^l_{\text{aux}})') \right), \quad (5.2.32)$$

with

$$v(\boldsymbol{H}) = \frac{\lambda_{\text{aux}} - d\boldsymbol{e}_0^{\mathrm{T}} \boldsymbol{H} \boldsymbol{e}_0}{\lambda_{\text{aux}}^2} \quad (5.2.33)$$

a function of the empirical auxiliary covariance $\boldsymbol{H} = \frac{1}{L}\sum_l \boldsymbol{r}^l_{\text{aux}} (\boldsymbol{r}^l_{\text{aux}})^{\mathrm{T}}$. Combining (5.2.31) and (5.2.32) the κ and $\hat{\kappa}$-independent version of (5.2.17) is then the self-consistency condition

$$\pi_\phi[\phi] = \sum_d \frac{p(d)d}{\bar{d}} \int \prod_{i=1}^{d-1} \mathcal{D}\phi^i \, \pi_\phi[\phi^i] \left\langle \delta\Big(\phi - \Phi[\hat{\Phi}[\phi^1], \ldots, \hat{\Phi}[\phi^{d-1}]]\Big) \right\rangle_\gamma, \quad (5.2.34)$$

for the distribution π_ϕ of the functions ϕ. Similar large-L saddle point arguments also yield for the generalisation error (5.2.21) (see Appendix B)

$$\epsilon_g = \lim_{\lambda \to 0} \sum_d p(d) \left\langle \int \prod_{i=1}^{d} \mathcal{D}\phi^i \pi_\phi[\phi^i] \int \mathrm{d}\boldsymbol{r} \prod_{l=1}^{L} \mathrm{d}\boldsymbol{r}^l_{\text{aux}} \left(\frac{1}{\gamma/\sigma^2 + \lambda} - \frac{v(\boldsymbol{H}) d \boldsymbol{e}_0^{\mathrm{T}} \boldsymbol{r} \boldsymbol{r}^{\mathrm{T}} \boldsymbol{e}_0}{(\gamma/\sigma^2 + \lambda)^2} \right) q_\phi(\boldsymbol{r}, \{\boldsymbol{r}^l_{\text{aux}}\}) \right\rangle_\gamma, \quad (5.2.35)$$

with

$$q_\phi(\boldsymbol{r}, \{\boldsymbol{r}^l_{\text{aux}}\}) \propto \exp\left(-\Phi[\hat{\Phi}[\phi^1], \ldots, \hat{\Phi}[\phi^d]](\boldsymbol{r}, \{\boldsymbol{r}^l_{\text{aux}}\}) \right). \quad (5.2.36)$$

Now that the large-L limit has been taken, an ansatz for solving the saddle point conditions (5.2.34) can be proposed. We take the 'energy functions' $\phi(\boldsymbol{r}, \{\boldsymbol{r}^l_{\text{aux}}\})$ to be of quadratic form again. In contrast to the case of global kernel normalisation, however, the

covariances of the $\boldsymbol{r}$ variables are chosen to depend on the local $\boldsymbol{r}_{\text{aux}}$ variables through the empirical auxiliary covariance $\boldsymbol{H} = \frac{1}{L}\sum_l \boldsymbol{r}^l_{\text{aux}}(\boldsymbol{r}^l_{\text{aux}})^{\mathrm{T}}$. This dependence is motivated by the definitions (5.2.31), (5.2.33) and (5.2.35) in which interactions between $\boldsymbol{r}$ and the auxiliary variables only appear via $\boldsymbol{H}$. Specifically, we write

$$\phi(\boldsymbol{r}, \{\boldsymbol{r}^l_{\text{aux}}\}) = \frac{1}{2}\boldsymbol{r}^{\mathrm{T}}[\boldsymbol{W}(\boldsymbol{H})]^{-1}\boldsymbol{r} + \frac{1}{2}\sum_l (\boldsymbol{r}^l_{\text{aux}})^{\mathrm{T}}\boldsymbol{V}^{-1}_{\text{aux}}\boldsymbol{r}^l_{\text{aux}}, \tag{5.2.37}$$

for $\boldsymbol{W}(\boldsymbol{H})$ a matrix function of the empirical covariance $\boldsymbol{H}$ of the $\boldsymbol{r}_{\text{aux}}$ variables.

Substituting the ansatz (5.2.37) into (5.2.32) we have

$$\begin{aligned}\hat{\Phi}[\phi'] = -\log \int \mathrm{d}\boldsymbol{r}' \prod_{l=1}^{L} \mathrm{d}(\boldsymbol{r}^l_{\text{aux}})' \exp\Big(&-\frac{1}{2}(\boldsymbol{r}')^{\mathrm{T}}[\boldsymbol{W}'(\boldsymbol{H}')]^{-1}\boldsymbol{r}' - \boldsymbol{r}^{\mathrm{T}}\boldsymbol{X}\boldsymbol{r}' \\ &- \frac{1}{2}\sum_l (\boldsymbol{r}^l_{\text{aux}})'^{\mathrm{T}}(\boldsymbol{V}'_{\text{aux}})^{-1}(\boldsymbol{r}^l_{\text{aux}})' - \sum_l (\boldsymbol{r}^l_{\text{aux}})^{\mathrm{T}}\boldsymbol{X}(\boldsymbol{r}^l_{\text{aux}})'\Big).\end{aligned} \tag{5.2.38}$$

Similar to the global calculation we wish to integrate (5.2.38) over $\boldsymbol{r}'$ and $\boldsymbol{r}'_{\text{aux}}$, but a direct attack is complicated because of the $\boldsymbol{r}'_{\text{aux}}$-dependence in $\boldsymbol{W}'$. However, we can think of the factors in the second line of (5.2.38) as defining a Gaussian weight on $\Delta^l \equiv (\boldsymbol{r}^l_{\text{aux}})'$, so that the Δ^l are L i.i.d. Gaussian random variables with distribution $\Delta^l \sim \mathcal{N}(-\boldsymbol{V}'_{\text{aux}}\boldsymbol{X}\boldsymbol{r}^l_{\text{aux}}, \boldsymbol{V}'_{\text{aux}})$. Discarding irrelevant additive constants then yields

$$\begin{aligned}\hat{\Phi}[\phi'] = -\log\Bigg[&\exp\Big(\frac{1}{2}\sum_l (\boldsymbol{r}^l_{\text{aux}})^{\mathrm{T}}\boldsymbol{X}\boldsymbol{V}'_{\text{aux}}\boldsymbol{X}\boldsymbol{r}^l_{\text{aux}}\Big)\int \mathrm{d}\boldsymbol{r}' \exp(-\boldsymbol{r}^{\mathrm{T}}\boldsymbol{X}\boldsymbol{r}') \\ &\left\langle \exp\left(-\frac{1}{2}(\boldsymbol{r}')^{\mathrm{T}}\Big[\boldsymbol{W}'\Big(\frac{1}{L}\sum_l \Delta^l(\Delta^l)^{\mathrm{T}}\Big)\Big]^{-1}\boldsymbol{r}'\right)\right\rangle_{\Delta^l}\Bigg].\end{aligned} \tag{5.2.39}$$

The key step is now that, in the limit of large L, the argument of $\boldsymbol{W}'$ becomes self averaging, so that we may rewrite (5.2.39) as

$$\hat{\Phi}[\phi'] = -\frac{1}{2}\sum_l (\boldsymbol{r}^l_{\text{aux}})^{\mathrm{T}}\boldsymbol{X}\boldsymbol{V}'_{\text{aux}}\boldsymbol{X}\boldsymbol{r}^l_{\text{aux}} - \frac{1}{2}\boldsymbol{r}^{\mathrm{T}}\boldsymbol{X}\boldsymbol{W}'(\boldsymbol{V}'_{\text{aux}} + \boldsymbol{V}'_{\text{aux}}\boldsymbol{X}\boldsymbol{H}\boldsymbol{X}\boldsymbol{V}'_{\text{aux}})\boldsymbol{X}\boldsymbol{r}. \tag{5.2.40}$$

With the integration in $\hat{\Phi}$ performed this way we may substitute (5.2.37) and (5.2.40)

into (5.2.34) to get the saddle point condition

$$\pi[\boldsymbol{V}_{\text{aux}}, \boldsymbol{W}(\cdot)] = \sum_d \frac{p(d)d}{\bar{d}} \int \prod_{i=1}^{d-1} \mathrm{d}\boldsymbol{V}_{\text{aux}}^i \mathcal{D}\boldsymbol{W}^i \pi[\boldsymbol{V}_{\text{aux}}^i, \boldsymbol{W}^i(\cdot)] \\ \left\langle \prod_{\boldsymbol{H}} \delta\Big(\boldsymbol{W}(\boldsymbol{H}) - [\tilde{\boldsymbol{O}}(\boldsymbol{H}) - \sum_{i=1}^{d-1} \boldsymbol{X}\boldsymbol{W}^i(\boldsymbol{V}_{\text{aux}}^i + \boldsymbol{V}_{\text{aux}}^i \boldsymbol{X}\boldsymbol{H}\boldsymbol{X}\boldsymbol{V}_{\text{aux}}^i)\boldsymbol{X}]^{-1}\Big) \right. \\ \left. \times \delta\Big(\boldsymbol{V}_{\text{aux}} - [\boldsymbol{O}_{\text{aux}} - \sum_{i=1}^{d-1} \boldsymbol{X}\boldsymbol{V}_{\text{aux}}^i \boldsymbol{X}]^{-1}\Big) \right\rangle_\gamma , \quad (5.2.41)$$

where we have defined $\boldsymbol{O}_{\text{aux}}$ in a similar manner to $\boldsymbol{O}$ (see (4.2.10)) but with κ equal to one and γ equal to zero, and

$$\tilde{\boldsymbol{O}}(\boldsymbol{H}) = d \left(\begin{array}{cccc|ccc} c_0 + \frac{1}{dv(\boldsymbol{H})(\gamma/\sigma^2+\lambda)} & \frac{1}{2}c_1 & \dots & \frac{1}{2}c_p & 0 & \dots & 0 \\ \frac{1}{2}c_1 & & & & -\mathrm{i} & & \\ \vdots & & & & & \ddots & \\ \frac{1}{2}c_p & & & & & & -\mathrm{i} \\ \hline 0 & -\mathrm{i} & & & & & \\ \vdots & & \ddots & & & \boldsymbol{0}_{p,p} & \\ 0 & & & -\mathrm{i} & & & \end{array} \right) . \quad (5.2.42)$$

We use $\prod_{\boldsymbol{H}}$ in (5.2.41) to represent a product over all possible arguments $\boldsymbol{H}$ of the matrix function $\boldsymbol{W}(\boldsymbol{H})$.

A similar large-L self-averaging argument can be made for the generalisation error, (5.2.35). Since each $\boldsymbol{r}_{\text{aux}}^l$ belongs to a Gaussian distribution with common covariance given by q_ϕ we may replace the empirical covariances $\boldsymbol{H} = \frac{1}{L}\sum_l \boldsymbol{r}_{\text{aux}}^l (\boldsymbol{r}_{\text{aux}}^l)^{\mathrm{T}}$ in (5.2.35) with $\boldsymbol{H}_m = (\boldsymbol{O}_{\text{aux}} - \sum_{i=1}^d \boldsymbol{X}\boldsymbol{V}_{\text{aux}}^i \boldsymbol{X})^{-1}$, the local auxiliary marginal. This simplifies (5.2.35) to give

$$\epsilon_g = \lim_{\lambda\to 0} \sum_d p(d) \left\langle \int \prod_{i=1}^d \mathrm{d}\boldsymbol{V}_{\text{aux}}^i \mathrm{d}\boldsymbol{W}^i(\cdot) \pi[\boldsymbol{V}_{\text{aux}}^i, \boldsymbol{W}^i(\cdot)] \right. \\ \left. \left(\frac{1}{\gamma/\sigma^2+\lambda} - \frac{v(\boldsymbol{H}_m) d \boldsymbol{e}_0^{\mathrm{T}} \boldsymbol{V}_m \boldsymbol{e}_0}{(\gamma/\sigma^2+\lambda)^2} \right) \right\rangle_\gamma , \quad (5.2.43)$$

with

$$\boldsymbol{V}_m = \Big[\tilde{\boldsymbol{O}}(\boldsymbol{H}_m) - \sum_{i=1}^d \boldsymbol{X}\boldsymbol{W}^i(\boldsymbol{V}_{\text{aux}}^i + \boldsymbol{V}_{\text{aux}}^i \boldsymbol{X}\boldsymbol{H}_m\boldsymbol{X}\boldsymbol{V}_{\text{aux}}^i)\boldsymbol{X}\Big]^{-1} . \quad (5.2.44)$$

It is worth comparing (5.2.43) to the corresponding expression (4.2.14) for the case of a globally normalised kernel. The normalisation factor κ there has been replaced by

$$v(\boldsymbol{H}_m) = \frac{\lambda_{\text{aux}} - d\boldsymbol{e}_0^{\mathrm{T}}\boldsymbol{H}_m\boldsymbol{e}_0}{\lambda_{\text{aux}}^2}. \tag{5.2.45}$$

This itself again looks like a contribution to a generalisation error with a constant normalisation factor, κ, equal to one and in the absence of examples, which is reassuring: for γ equal to zero (no training examples), the generalisation error is just the prior variance. So $v(\boldsymbol{H}_m)$ is the local prior variance of the auxiliary variables, and it is this quantity that should provide the normalisation factor for the original variables.

The inverse matrix in (4.2.14) corresponds to $\boldsymbol{V}_m$ in (5.2.43) and in both cases is the covariance matrix of $\boldsymbol{r}$. In the locally normalised scenario, the large L-limit tells us that the matrix function $\boldsymbol{W}(\,\cdot\,)$ is evaluated at a definite point, namely the local auxiliary marginal, $\boldsymbol{H}_m$.

Conceptually, we are now in principle done: we have taken the large L limit and have a self-consistency condition for the joint distribution of $\boldsymbol{V}_{\text{aux}}$ and $\boldsymbol{W}(\,\cdot\,)$. But we have paid for this by the fact that $\boldsymbol{W}(\,\cdot\,)$ is an entire matrix function. This would be very difficult to parameterise for a numerical solution of (5.2.41) by population dynamics. Our final step in the analysis is therefore to reduce the description to one in terms of matrices rather than matrix functions.

To motivate this, we recall the cavity interpretation of the update equations given in Chapter 4. For a vertex with degree d we sample $d-1$ neighbours independently and combine their cavity distributions with information about the current vertex to create a cavity distribution or 'message' to send to the d-th neighbour. In the case of local normalisation, the $d-1$ neighbours each pass an auxiliary cavity covariance $\boldsymbol{V}_{\text{aux}}$ and a cavity covariance function $\boldsymbol{W}(\boldsymbol{H})$. Auxiliary covariances are combined with local information in direct analogy to (5.1.38) for globally normalised kernels to create a new auxiliary covariance according to $\boldsymbol{V}_{\text{aux}} = (\boldsymbol{O}_{\text{aux}} - \sum_{i=1}^{d-1} \boldsymbol{X}\boldsymbol{V}_{\text{aux}}^i\boldsymbol{X})^{-1}$. The covariance function $\boldsymbol{W}(\boldsymbol{H})$ is calculated slightly differently, as specified by the functional delta function in the second line of (5.2.41), due to its dependence on the auxiliary covariance through its argument. It combines incoming function covariances $\boldsymbol{W}^i$ evaluated at $\boldsymbol{V}_{\text{aux}}^i + \boldsymbol{V}_{\text{aux}}^i\boldsymbol{X}\boldsymbol{H}\boldsymbol{X}\boldsymbol{V}_{\text{aux}}^i$ with local information dependent on the empirical covariance $\boldsymbol{H}$ to calculate the outgoing cavity covariance function $\boldsymbol{W}(\boldsymbol{H})$.

The important point is now that when we come to calculate the generalisation error, we do not require the full functions $\tilde{\boldsymbol{O}}(\,\cdot\,)$ and $\boldsymbol{W}^i(\,\cdot\,)$, but only their values at specific arguments. The marginal covariance in $\tilde{\boldsymbol{O}}(\boldsymbol{H}_m)$ can be written as

$$\boldsymbol{H}_m = (\boldsymbol{O}_{\text{aux}} - \sum_{i=1}^{d} \boldsymbol{X}\boldsymbol{V}_{\text{aux}}^i\boldsymbol{X})^{-1} = (\boldsymbol{V}_{\text{aux}}^{-1} - \boldsymbol{X}\overleftarrow{\boldsymbol{V}}_{\text{aux}}\boldsymbol{X})^{-1}, \tag{5.2.46}$$

where $\boldsymbol{V}_{\text{aux}} = (\boldsymbol{O}_{\text{aux}} - \sum_{i=1}^{d-1} \boldsymbol{X}\boldsymbol{V}_{\text{aux}}^i\boldsymbol{X})^{-1}$ is the auxiliary covariance message that would be sent to the d-th neighbour, and $\overleftarrow{\boldsymbol{V}}_{\text{aux}} \equiv \boldsymbol{V}_{\text{aux}}^d$ is the *reverse message* that this vertex sends. This form of argument is maintained consistently at neighbouring vertices in the self-consistency condition (5.2.41). Indeed, substituting $\boldsymbol{H}_m$ into the argument of $\boldsymbol{W}^i$ on the RHS of (5.2.41) we see that, by application of Woodbury's identity, this may be rewritten as

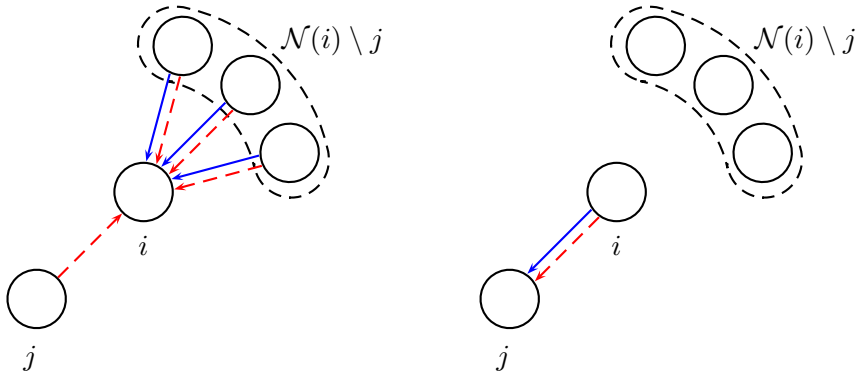
$$\begin{aligned}
&\boldsymbol{V}_{\text{aux}}^i + \boldsymbol{V}_{\text{aux}}^i\boldsymbol{X}\boldsymbol{H}_m\boldsymbol{X}\boldsymbol{V}_{\text{aux}}^i \\
&\quad = \boldsymbol{V}_{\text{aux}}^i + \boldsymbol{V}_{\text{aux}}^i\boldsymbol{X}\Big(\boldsymbol{O}_{\text{aux}} - \sum_{j=1}^{d-1}\boldsymbol{X}\boldsymbol{V}_{\text{aux}}^j\boldsymbol{X} - \boldsymbol{X}\overleftarrow{\boldsymbol{V}}_{\text{aux}}\boldsymbol{X}\Big)^{-1}\boldsymbol{X}\boldsymbol{V}_{\text{aux}}^i \\
&\quad = \Big[(\boldsymbol{V}_{\text{aux}}^i)^{-1} - \boldsymbol{X}\Big(\boldsymbol{O}_{\text{aux}} - \sum_{j\neq i}\boldsymbol{X}\boldsymbol{V}_{\text{aux}}^j\boldsymbol{X} - \boldsymbol{X}\overleftarrow{\boldsymbol{V}}_{\text{aux}}\boldsymbol{X}\Big)^{-1}\boldsymbol{X}\Big]^{-1}.
\end{aligned} \tag{5.2.47}$$

This is of the same form as (5.2.46), and has the analogous interpretation of the marginal auxiliary covariance at vertex i, $\boldsymbol{H}_m^i$. It is expressed again in terms of a message this vertex sends, $\boldsymbol{V}_{\text{aux}}^i$, and a reverse message sent to this vertex. The latter is $\overleftarrow{\boldsymbol{V}}_{\text{aux}}^i = (\boldsymbol{O}_{\text{aux}} - \sum_{j\neq i}\boldsymbol{X}\boldsymbol{V}_{\text{aux}}^j\boldsymbol{X} - \boldsymbol{X}\overleftarrow{\boldsymbol{V}}_{\text{aux}}\boldsymbol{X})^{-1}$ and is constructed from information received from all other neighbours of the central vertex, including the d-th one. The cavity interpretation is summarised in figure 5.1.

The discussion so far suggests that we should consider a reduced but conditional distribution of messages where the covariance function $\boldsymbol{W}(\,\cdot\,)$ is evaluated only at the local marginal auxiliary covariance:

$$\pi[\boldsymbol{V}_{\text{aux}}, \boldsymbol{V}|\overleftarrow{\boldsymbol{V}}_{\text{aux}}] = \int \mathcal{D}\boldsymbol{W}\pi[\boldsymbol{V}_{\text{aux}}, \boldsymbol{W}(\,\cdot\,)]\delta\Big(\boldsymbol{V} - \boldsymbol{W}([\boldsymbol{V}_{\text{aux}}^{-1} - \boldsymbol{X}\overleftarrow{\boldsymbol{V}}_{\text{aux}}\boldsymbol{X}]^{-1})\Big). \tag{5.2.48}$$

Substituting this into (5.2.41), we can now solve the problem of passing the full functions $\boldsymbol{W}$. After some algebra, including applying (5.2.47), we get the following self-consistency



(a) Messages being received from neighbours

(b) Update sent to receiving vertex

Figure 5.1: A diagram explaining the cavity interpretation of the 'counterflow' updates for the locally normalised kernel. (a) shows vertex i receiving covariance messages $\boldsymbol{V}$ from all the neighbours except the receiving vertex j (represented by solid arrows) and receiving covariance messages $\boldsymbol{V}_{\text{aux}}$ from all neighbours *including* the receiving vertex (represented by dashed arrows). (b) shows that after the new updates are calculated using the incoming covariances, the receiving vertex has *both* $\boldsymbol{V}$ and $\boldsymbol{V}_{\text{aux}}$ updated.

equation:

$$\pi[\boldsymbol{V}_{\rm aux},\boldsymbol{V}|\overleftarrow{\boldsymbol{V}}_{\rm aux}] = \sum_d \frac{p(d)d}{\bar{d}} \int \prod_{i=1}^{d-1} \mathrm{d}\boldsymbol{V}^i_{\rm aux}\mathrm{d}\boldsymbol{V}^i\mathrm{d}\overleftarrow{\boldsymbol{V}}^i_{\rm aux}\,\pi[\boldsymbol{V}^i_{\rm aux},\boldsymbol{V}^i|\overleftarrow{\boldsymbol{V}}^i_{\rm aux}] \left\langle \delta\Big(\boldsymbol{V}_{\rm aux} - \Big[\boldsymbol{O}_{\rm aux} - \sum_{i=1}^{d-1}\boldsymbol{X}\boldsymbol{V}^i_{\rm aux}\boldsymbol{X}\Big]^{-1}\Big)\delta\Big(\boldsymbol{V} - \Big[\tilde{\boldsymbol{O}} - \sum_{i=1}^{d-1}\boldsymbol{X}\boldsymbol{V}^i\boldsymbol{X}\Big]^{-1}\Big) \times \prod_{i=1}^{d-1}\delta\Big(\overleftarrow{\boldsymbol{V}}^i_{\rm aux} - \Big[\boldsymbol{O}_{\rm aux} - \sum_{j\neq i}\boldsymbol{X}\boldsymbol{V}^j_{\rm aux}\boldsymbol{X} - \boldsymbol{X}\overleftarrow{\boldsymbol{V}}_{\rm aux}\boldsymbol{X}\Big]^{-1}\Big)\right\rangle_\gamma . \quad (5.2.49)$$

where we have abbreviated $\tilde{\boldsymbol{O}} = \tilde{\boldsymbol{O}}([\boldsymbol{V}^{-1}_{\rm aux} - \boldsymbol{X}\overleftarrow{\boldsymbol{V}}_{\rm aux}\boldsymbol{X}]^{-1})$. The generalisation error (5.2.43) can be rewritten in the same fashion as

$$\epsilon_g = \lim_{\lambda\to 0}\sum_d p(d)\left\langle \int \prod_{i=1}^{d}\mathrm{d}\boldsymbol{V}^i_{\rm aux}\mathrm{d}\boldsymbol{V}^i\mathrm{d}\overleftarrow{\boldsymbol{V}}^i_{\rm aux}\,\pi[\boldsymbol{V}^i_{\rm aux},\boldsymbol{V}^i|\overleftarrow{\boldsymbol{V}}^i_{\rm aux}] \left(\frac{1}{\gamma/\sigma^2+\lambda} - \frac{v(\boldsymbol{H}_m)d\boldsymbol{e}_0^{\rm T}\boldsymbol{V}_m\boldsymbol{e}_0}{(\gamma/\sigma^2+\lambda)^2}\right) \times \prod_{i=1}^{d}\delta\Big(\overleftarrow{\boldsymbol{V}}^i_{\rm aux} - \Big[\boldsymbol{O}_{\rm aux} - \sum_{j\neq i}\boldsymbol{X}\boldsymbol{V}^j_{\rm aux}\boldsymbol{X} - \boldsymbol{X}\overleftarrow{\boldsymbol{V}}_{\rm aux}\boldsymbol{X}\Big]^{-1}\Big)\right\rangle_\gamma . \quad (5.2.50)$$

Both here and in the self-consistency equation (5.2.49), the delta functions for the reverse messages effectively ensure that these messages are consistent with the forward messages $\boldsymbol{V}^i_{\rm aux}$ and with $\overleftarrow{\boldsymbol{V}}_{\rm aux}$.

To find $\pi[\boldsymbol{V}_{\rm aux},\boldsymbol{V}|\overleftarrow{\boldsymbol{V}}_{\rm aux}]$ numerically from (5.2.49) would require conditional population dynamics [see 43], which is still rather challenging. We therefore now make an approximation. Since $\overleftarrow{\boldsymbol{V}}_{\rm aux}$ is a message from the d-th neighbour like the other auxiliary covariances $\boldsymbol{V}^i_{\rm aux}$ being sent from neighbours $i = 1,\ldots,d-1$, it has probability weight $\pi[\overleftarrow{\boldsymbol{V}}_{\rm aux}]$. Multiplying (5.2.49) by this weight and integrating over $\overleftarrow{\boldsymbol{V}}_{\rm aux}$ gives on the LHS the unconditional distribution $\pi[\boldsymbol{V}_{\rm aux},\boldsymbol{V}]$. Our approximation consists of dropping at this stage the conditioning also on the RHS, replacing $\pi[\boldsymbol{V}^i_{\rm aux},\boldsymbol{V}^i|\overleftarrow{\boldsymbol{V}}^i_{\rm aux}]$ by $\pi[\boldsymbol{V}^i_{\rm aux},\boldsymbol{V}^i]$. The reverse messages can then be integrated out, leading to the approximate self-consistency equation

$$\pi[\boldsymbol{V}_{\rm aux},\boldsymbol{V}] = \sum_d \frac{p(d)d}{\bar{d}} \int \prod_{i=1}^{d-1}\mathrm{d}\boldsymbol{V}^i_{\rm aux}\mathrm{d}\boldsymbol{V}^i\pi[\boldsymbol{V}^i_{\rm aux},\boldsymbol{V}^i]\int \mathrm{d}\overleftarrow{\boldsymbol{V}}_{\rm aux}\pi[\overleftarrow{\boldsymbol{V}}_{\rm aux}] \left\langle \delta\Big(\boldsymbol{V}_{\rm aux} - \Big[\boldsymbol{O}_{\rm aux} - \sum_{i=1}^{d-1}\boldsymbol{X}\boldsymbol{V}^i_{\rm aux}\boldsymbol{X}\Big]^{-1}\Big)\delta\Big(\boldsymbol{V} - \Big[\tilde{\boldsymbol{O}} - \sum_{i=1}^{d-1}\boldsymbol{X}\boldsymbol{V}^i\boldsymbol{X}\Big]^{-1}\Big)\right\rangle_\gamma . \quad (5.2.51)$$

This is exactly (4.2.26) which we derived in Chapter 4. We see that from the replica derivation the approximation we have made in Chapter 4 becomes clear; we have ignored consistency constraints when calculating the auxiliary marginals.

Equation (5.2.51) has two apparent singularities, one in $\tilde{\boldsymbol{O}}$ as λ approaches zero in the absence of local examples γ, and another one in $v(\boldsymbol{H}_m)$ (which appears in the definition of $\tilde{\boldsymbol{O}}$) as λ_{aux} approaches zero. Apparent divergences in $\tilde{\boldsymbol{O}}$ are resolved in a similar manner to those in section 5.1. Similar problems in $v(\boldsymbol{H}_m)$ as λ_{aux} approaches zero can be avoided by introducing $\boldsymbol{M}_{\text{aux},d} = \boldsymbol{O}_{\text{aux}} - \sum_{i=1}^{d} \boldsymbol{X}\boldsymbol{V}_{\text{aux}}^{i}\boldsymbol{X} - \boldsymbol{e}_0 \frac{d}{\lambda_{\text{aux}}} \boldsymbol{e}_0^{\mathrm{T}}$ and applying the Woodbury identity to give $v(\boldsymbol{H}_m) = (d\boldsymbol{e}_0^{\mathrm{T}} \boldsymbol{M}_{\text{aux},d}^{-1} \boldsymbol{e}_0)^{-1}$ in the limit λ_{aux} equal to zero.

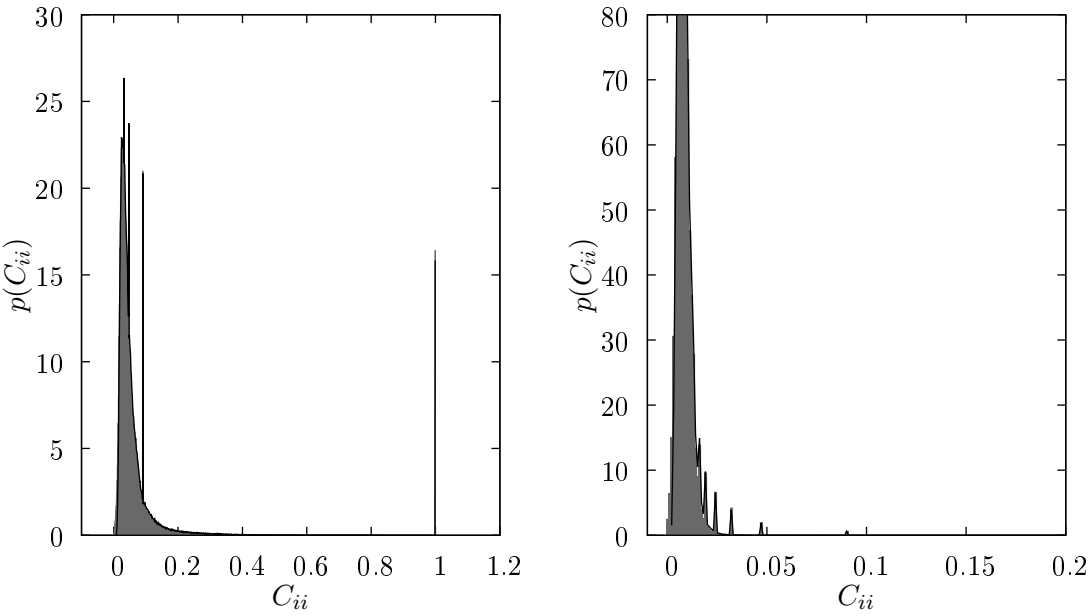
The approximation of the generalisation error (5.2.50) that corresponds to the approximate self-consistency equation (5.2.51) reads, after again applying the Woodbury formula to deal with the two apparent divergences as λ and λ_{aux} approach zero,

$$\epsilon_g = \sum_d p(d) \left\langle \int \prod_{i=1}^{d} \mathrm{d}\boldsymbol{V}_{\text{aux}}^{i} \mathrm{d}\boldsymbol{V}^{i}\, \pi[\boldsymbol{V}_{\text{aux}}^{i}, \boldsymbol{V}^{i}] \frac{1}{\gamma/\sigma^2 + (\boldsymbol{e}_0^{\mathrm{T}} \boldsymbol{M}_{\text{aux},d}^{-1} \boldsymbol{e}_0)^{-1} \boldsymbol{e}_0^{\mathrm{T}} \boldsymbol{M}_d^{-1} \boldsymbol{e}_0} \right\rangle_{\gamma} \tag{5.2.52}$$

with $\boldsymbol{M}_d$ defined as in Chapter 4. Equation (5.2.52) is the analogue of (4.2.28) and can be evaluated straightforwardly once a population estimate for $\pi[\boldsymbol{V}_{\text{aux}}, \boldsymbol{V}]$ has been found. We see that the cavity equations that we postulated in Chapter 4 would calculate the approximation of the locally normalised kernel can be calculated more explicitly using the replica method. As is typical in cases where both the replica and cavity methods can be used to calculate a solution we see that, the cavity method in Chapter 4 gave us an intuitive understanding of the problem, but the power of the replica method allowed us to truly understand the details of the problem.

The interpretation of (5.2.52) is similar to (5.1.39): information from the rest of the graph provides an effective prior variance at any given vertex; the new element here is that this is normalised by the prior variance in the absence of data which must be computed from the auxiliary variables.

As was stated in Chapter 4 it is difficult to asses a priori the quality of the approximation we have made above, but numerical results (see figure 4.4) show that in practice it is very accurate, surprisingly so, even for graph ensembles with a large spread of unnormalised prior variances. On this basis one might speculate whether the approximation could



(a) $\nu = 1.172$ (b) $\nu = 6.210$

Figure 5.2: (a) Grey: histogram of posterior variances at $\nu = 1.172$ for the locally normalised random walk kernel with $a = 2$, $p = 10$, averaged over ten samples each of teacher functions, data and Erdős-Rényi graphs with mean degree $\lambda = 3$ and $V = 1000$ vertices. Black: cavity prediction for this distribution in the large graph limit. (b) As (a) but for $\nu = 6.210$.

in fact be exact. This is further substantiated by looking not just at the average of the posterior variance over vertices, which is the Bayes error, but its distribution across vertices. As shown in figure 5.2, the cavity predictions for this distribution are in very good agreement with the results of numerical simulations. This holds not only for the two values of ν shown, but along the entire learning curve. To show that this is the case analytically remains an open question.

5.3 Summary

In this chapter we have derived, through a different method, the result we obtained using the cavity method in Chapter 4. Doing this gave us deeper insight into the approximations we proposed in the cavity chapter in order to solve the learning curves for locally normalised random walk kernels. We have shown that this approximation

amounts to not requiring consistency between neighbouring auxiliary cavity covariances when constructing the full marginal.

To derive this result we began by calculating the learning curve for the case of a globally normalised kernel. Similar to Chapter 4 we focussed on the partition function since from this we could generate the learning curve. We began by rewriting the replicated partition function in terms of a single effective site formulation. The aim of this was to be able to apply a saddle point approximation in the limit of large graphs. To get our partition function into this form we had to introduce replica densities that measured the density of states in the replicated system. In order to derive the earlier cavity result, we then assumed replica symmetry. We showed that after assuming replica symmetry and taking the saddle point prediction we indeed ended up with the approximation we derived using the cavity method in the previous chapter.

After deriving the cavity result using replicas for the globally normalised kernel we then used this experience to derive the learning curves for locally normalised kernel. This was the main focus of this chapter. In the cavity chapter we had to 'guess' at the correct structure of the self-consistent equation and learning curve. By deriving it using replicas we wanted to take a more principled approach.

We began by introducing delta enforcement terms for the normalising constants into the partition function. With the introduction of these enforcement terms we were no longer able to directly apply the method we had used in the globally normalised case. We showed that this could be resolved and we could rewrite this once more into a form that initially we could solve using the same technique we applied for the globally normalised kernel. From this point we could initially progress along the same path as for the global case until we derived the self consistent equations for the case of the locally normalised kernel. Here we found that further approximations were required, which included a second saddle point approximation *inside* the self consistent equations. We found that once this saddle point approximation had been made we could again solve the self-consistent equations. This time, however, there were complicated relationships between some of the variables. In particular we found that the variables responsible for calculating the final variance of the GP were coupled in a highly non-trivial way to the variables responsible for calculating the normaliser. This complicated relationship

was simplified by self-averaging arguments and 'working backwards' from the generalisation error approximation. Finally we found that, after some lengthy calculation, we could once more write the generalisation error and self-consistent equations so that they resembled the cavity equations derived in the previous chapter but with an additional consistency constraint between incoming auxiliary cavity covariances. By ignoring these consistency constraints we retrieved the equations obtained in Chapter 4.

Now that we have accurate approximations for the generalisation error of a GP with a random walk kernel for both local and global normalisations in a matched scenario we will consider the more realistic case of model mismatch. We will see in the next chapter that this complicates the equations greatly and introduces a substantial number of additional variables over which our cavity equations are defined. We will focus on the case where both student and teacher have a random walk kernel but with potentially different hyperparamters.

CHAPTER 6

PREDICTING THE GENERALISATION ERROR WITH MODEL MISMATCH

The generalisation error of a learning algorithm within a matched scenario does not necessarily give a good indication of performance in real applications. This was touched upon in section 3.3 when we compared the performance of the two kernel normalisations against one another to decide which was probabilistically more plausible. In this chapter we will derive a prediction of the generalisation error of a GP in the more realistic scenario of model mismatch. This will follow a similar (but more complicated) path to that used to calculate the matched generalisation error in Chapter 4. In general the teacher's prior GP could have a completely different kernel. We will assume in this chapter, however, that the teacher has a random walk kernel with potentially different hyperparameters.

6.1 The generalisation error with model mismatch

The aim of this chapter is to once more study GP regression on a graph for functions defined on the vertices of the graph, but in the presence of mismatch. We assume that we are presented inputs $\boldsymbol{X} = (x_1, \ldots, x_N)^{\mathrm{T}}$ with corresponding noisy outputs $\boldsymbol{y} = (y_1, \ldots, y_N)^{\mathrm{T}}$ from a function we wish to estimate.

We focus on the random walk kernel given by

$$\hat{\boldsymbol{C}} = (\boldsymbol{I} - a^{-1}\boldsymbol{L})^p = (\boldsymbol{I}(1 - a^{-1}) + a^{-1}\boldsymbol{D}^{1/2}\boldsymbol{A}\boldsymbol{D}^{1/2})^p \tag{6.1.1}$$

which we will assume is normalised using $\mathcal{K} = \mathrm{diag}(\kappa_1, \ldots, \kappa_V)$. We will derive the cavity approximation solely for global normalisation[1] so that $\kappa_i = V^{-1}\sum_j \hat{C}_{jj}$ for all i. With the kernel (which is really a $V \times V$ matrix) given by $\boldsymbol{C} = \mathcal{K}^{-1/2}\hat{\boldsymbol{C}}\mathcal{K}^{-1/2}$ the GP prior over functions on a graph can be written explicitly as

$$P(\boldsymbol{f}) \propto \exp\left(-\frac{1}{2}\boldsymbol{f}^{\mathrm{T}}(\mathcal{K}_f^{-1/2}\hat{\boldsymbol{C}}_f\mathcal{K}_f^{-1/2})^{-1}\boldsymbol{f} \right) \tag{6.1.2}$$

where, since f is on a graph, we have defined f via a vector $\boldsymbol{f} = (f(1), \ldots, f(V))^{\mathrm{T}}$ and where we have used a subscript f to indicate explicitly that the covariance is for functions, f.

Combining the prior distribution with a Gaussian likelihood distribution with variance σ_f^2 we arrive, via Bayes' theorem, at the posterior distribution. This is simply another GP given by

$$P(\boldsymbol{f}|\boldsymbol{X}, \boldsymbol{y}) = \frac{\exp\left(-\frac{1}{2}\boldsymbol{f}^{\mathrm{T}}\boldsymbol{C}_f^{-1}\boldsymbol{f} - \frac{1}{2\sigma_f^2}\sum_{\mu=1}^{N}(f_{x_\mu} - y_\mu)^2 \right)}{\int \mathrm{d}\boldsymbol{f}' \exp\left(-\frac{1}{2}(\boldsymbol{f}')^{\mathrm{T}}\boldsymbol{C}_f^{-1}\boldsymbol{f}' - \frac{1}{2\sigma_f^2}\sum_{\mu=1}^{N}(f'_{x_\mu} - y_\mu)^2 \right)} \tag{6.1.3}$$

We wish to characterise the performance of the GP. In a similar manner to the previous chapters we will use the generalisation error to do this. We will assume that the *student* GP is presented data generated by uniformly sampling across all vertices N values from a function g, corrupted by i.i.d. Gaussian noise with variance σ_g^2. Furthermore, we will assume that the function g was created by sampling from another *teacher* GP with a

[1] The extension of the method we will derive below for predicting the generalisation error for the locally normalised case discussed in chapters 3, 4 and 5 is trivial. It follows exactly the same prescription in Chapter 4 and its inclusion only masks the fundamental steps in the mismatch derivation.

possibly different kernel function $\boldsymbol{C}_g$. Finally for the specific case of GPs on random graphs we will assume the graph on which the function g was generated - and which $\boldsymbol{f}$ is being learned - is a sample from a random graph ensemble $\mathcal{G}$.

Under these assumptions we see that we must calculate the mean square error of the student GP over all possible data sets, teacher functions $\boldsymbol{g}$ and graphs G sampled from their respective distributions. This is known as the generalisation error (see section 2.1.4.2) and is given by

$$\epsilon_g(N) = \left\langle \left\langle \left\langle \left\langle \frac{1}{V} \sum_{i=1}^{V} (\langle f_i \rangle_{f|\boldsymbol{y},\boldsymbol{X}} - g_i)^2 \right\rangle_{\boldsymbol{y}|g,\boldsymbol{X}} \right\rangle_{g} \right\rangle_{\boldsymbol{X}} \right\rangle_{\mathcal{G}}. \tag{6.1.4}$$

where once more we have used the fact that we are on a graph to replace the functions f and g by V dimensional vectors $\boldsymbol{f}$ and $\boldsymbol{g}$ respectively.

Equation (6.1.4) is hard to calculate exactly. We will therefore now derive an approximation of (6.1.4) using the cavity method. In a similar manner to Chapter 4 we will first need to rewrite (6.1.4) in terms of a generating partition function. Since we will be using two GPs with random walk kernels with potentially different parameters, for the remainder of this chapter we will denote kernel hyper-parameters belonging to the teacher by a subscript g and those belonging to the student by a subscript f.

6.2 Creating the generating partition function

We derive the generating partition functional form of (6.1.4) in a similar manner to section 4.2.1. However, since the terms inside the average are no longer the posterior variance of the student GP, we will not be able to shift by the posterior mean to simplify the equations.

We begin by first examining the joint probability distribution of the teacher GP, $P(\boldsymbol{g})$, the data outputs given the inputs, $P(\boldsymbol{y}|\boldsymbol{X})$, and the student posterior GP, $P(\boldsymbol{f}|\boldsymbol{X},\boldsymbol{y})$. We see that by basic properties of probability distributions

$$P(\boldsymbol{f},\boldsymbol{y},\boldsymbol{g}|\boldsymbol{X}) = P(\boldsymbol{f}|\boldsymbol{X},\boldsymbol{y})P(\boldsymbol{y}|\boldsymbol{g},\boldsymbol{X})P(\boldsymbol{g}) \tag{6.2.1}$$

and that the generalisation error can be written as

$$\epsilon_g(N) = \frac{1}{V} \sum_{i=1}^{V} \left\langle \left\langle \int \mathrm{d}\boldsymbol{f}\, \mathrm{d}\boldsymbol{g}\, \mathrm{d}\boldsymbol{y}\, (f_i - g_i)^2 P(\boldsymbol{f},\boldsymbol{y},\boldsymbol{g}|\boldsymbol{X}) \right\rangle_{\boldsymbol{X}} \right\rangle_{\mathcal{G}}. \tag{6.2.2}$$

We therefore focus on creating a generating partition function using the joint distribution (6.2.1).

To calculate the joint distribution explicitly we require the explicit forms of the three distributions on the RHS of (6.2.1). The last two distributions come straight from our definitions and are given by

$$P(\boldsymbol{y}|\boldsymbol{g},\boldsymbol{X}) = \prod_{\mu=1}^{N} \frac{1}{\sqrt{2\pi\sigma_g^2}} \exp\Big(-\frac{1}{2\sigma_g^2}(y_\mu - g_{x_\mu})^2 \Big)$$
$$= \frac{1}{(2\pi\sigma_g^2)^{N/2}} \exp\Big(-\frac{1}{2\sigma_g^2}(\boldsymbol{y}-\boldsymbol{g}_{\boldsymbol{X}})^{\mathrm{T}}(\boldsymbol{y}-\boldsymbol{g}_{\boldsymbol{X}}) \Big) \quad (6.2.3)$$

with $\boldsymbol{g}_{\boldsymbol{X}} = (g(x_1),\ldots,g(x_N))^{\mathrm{T}}$ and

$$P(\boldsymbol{g}) = \frac{1}{(2\pi)^{V/2}\det^{1/2}(\boldsymbol{C}_g)} \exp\big(-\frac{1}{2}\boldsymbol{g}^{\mathrm{T}}\boldsymbol{C}_g^{-1}\boldsymbol{g} \big) \quad (6.2.4)$$

respectively. The first term in (6.2.1), however, cannot be as simply expressed. To calculate this we must apply Bayes' theorem (see (2.1.2)) to rewrite the posterior GP, $P(\boldsymbol{f}|\boldsymbol{X},\boldsymbol{y})$, in terms of distributions we know explicitly. We see that

$$P(\boldsymbol{f}|\boldsymbol{X},\boldsymbol{y}) = \frac{P(\boldsymbol{y}|\boldsymbol{f},\boldsymbol{X})P(\boldsymbol{f})}{P(\boldsymbol{y}|\boldsymbol{X})} \quad (6.2.5)$$
$$= D\frac{\exp\big(-\frac{1}{2\sigma_f^2}(\boldsymbol{y}-\boldsymbol{f}_{\boldsymbol{X}})^{\mathrm{T}}(\boldsymbol{y}-\boldsymbol{f}_{\boldsymbol{X}}) \big)\exp\big(-\frac{1}{2}\boldsymbol{f}^{\mathrm{T}}\boldsymbol{C}_f^{-1}\boldsymbol{f} \big)}{\exp\big(-\frac{1}{2}\boldsymbol{y}^{\mathrm{T}}\boldsymbol{K}_f^{-1}\boldsymbol{y} \big)} \quad (6.2.6)$$

where the denominator is the marginal likelihood given in (2.1.7), $(\boldsymbol{K}_f)_{\mu,\nu} = C_{x_\mu,x_\nu} + \delta_{\mu,\nu}\sigma_f^2$, $\boldsymbol{f}_{\boldsymbol{X}} = (f(x_1),\ldots,f(x_N))^{\mathrm{T}}$ and,

$$D = \frac{\det^{1/2}(\boldsymbol{K}_f)}{(2\pi)^{V/2}\sigma_f^N\det^{1/2}(\boldsymbol{C}_f)} \quad (6.2.7)$$

Combining these terms we see that the joint distribution is given by

$$P(\boldsymbol{f},\boldsymbol{y},\boldsymbol{g}|\boldsymbol{X}) = D'\exp\Bigg(-\frac{1}{2}\boldsymbol{f}^{\mathrm{T}}\boldsymbol{C}_f^{-1}\boldsymbol{f} - \frac{1}{2\sigma_f^2}(\boldsymbol{y}-\boldsymbol{f}_{\boldsymbol{X}})^{\mathrm{T}}(\boldsymbol{y}-\boldsymbol{f}_{\boldsymbol{X}})$$
$$- \frac{1}{2\sigma_g^2}(\boldsymbol{y}-\boldsymbol{g}_{\boldsymbol{X}})^{\mathrm{T}}(\boldsymbol{y}-\boldsymbol{g}_{\boldsymbol{X}}) - \frac{1}{2}\boldsymbol{g}^{\mathrm{T}}\boldsymbol{C}_g^{-1}\boldsymbol{g} + \frac{1}{2}\boldsymbol{y}^{\mathrm{T}}\boldsymbol{K}_f^{-1}\boldsymbol{y} \Bigg) \quad (6.2.8)$$

with

$$D' = \frac{\det^{1/2}(\boldsymbol{K}_f)}{(2\pi)^{V+N/2}\sigma_f^N\sigma_g^N\det^{1/2}(\boldsymbol{C}_f)\det^{1/2}(\boldsymbol{C}_g)}. \quad (6.2.9)$$

Now that we have (6.2.1) in the explicit form given by (6.2.8) we see that in a similar manner to section 4.2.1 we may rewrite the generalisation error as

$$\epsilon_g = -\frac{2}{V}\lim_{\lambda\to 0}\sum_{i=1}^{V}\left\langle \frac{\partial}{\partial\lambda}\log Z\right\rangle_{\boldsymbol{X},\mathcal{G}} \tag{6.2.10}$$

with

$$\begin{aligned} Z = \int \mathrm{d}\boldsymbol{f}\,\mathrm{d}\boldsymbol{y}\,\mathrm{d}\boldsymbol{g}\,\exp\Bigg(&-\frac{1}{2}\boldsymbol{f}^{\mathrm{T}}\boldsymbol{C}_f^{-1}\boldsymbol{f} - \frac{1}{2\sigma_f^2}(\boldsymbol{y}-\boldsymbol{f}_{\boldsymbol{X}})^{\mathrm{T}}(\boldsymbol{y}-\boldsymbol{f}_{\boldsymbol{X}}) \\ &- \frac{1}{2\sigma_g^2}(\boldsymbol{y}-\boldsymbol{g}_{\boldsymbol{X}})^{\mathrm{T}}(\boldsymbol{y}-\boldsymbol{g}_{\boldsymbol{X}}) - \frac{1}{2}\boldsymbol{g}^{\mathrm{T}}\boldsymbol{C}_g^{-1}\boldsymbol{g} + \frac{1}{2}\boldsymbol{y}^{\mathrm{T}}\boldsymbol{K}_f^{-1}\boldsymbol{y} \\ &- \frac{\lambda}{2}(\boldsymbol{f}-\boldsymbol{g})^{\mathrm{T}}(\boldsymbol{f}-\boldsymbol{g})\Bigg). \end{aligned} \tag{6.2.11}$$

It is worth noting that in (6.2.11) we have dropped the normalising terms from (6.2.8) because the generating term will create these (in the form $1/Z$) when it is differentiated.

We now proceed as we did in Chapter 4, and rewrite (6.2.11) into the form of a graphical model with single vertex and nearest neighbour interaction terms only.

6.3 Deriving the graphical model form

The first step in deriving the graphical model form of (6.2.11) is to marginalise over $\boldsymbol{y}$. Initially one might be tempted to do this directly: grouping together the terms involving the outputs, $\boldsymbol{y}$, we see that (6.2.11) may be rewritten as

$$\begin{aligned} Z = \int \mathrm{d}\boldsymbol{f}\mathrm{d}\boldsymbol{g}\,\exp\Bigg(&-\frac{1}{2}\boldsymbol{f}^{\mathrm{T}}\boldsymbol{C}_f^{-1}\boldsymbol{f} - \frac{1}{2\sigma_f^2}\boldsymbol{f}_{\boldsymbol{X}}{}^{\mathrm{T}}\boldsymbol{f}_{\boldsymbol{X}} - \frac{1}{2\sigma_g^2}\boldsymbol{g}_{\boldsymbol{X}}{}^{\mathrm{T}}\boldsymbol{g}_{\boldsymbol{X}} \\ &- \frac{1}{2}\boldsymbol{g}^{\mathrm{T}}\boldsymbol{C}_g^{-1}\boldsymbol{g} - \frac{\lambda}{2}(\boldsymbol{f}-\boldsymbol{g})^{\mathrm{T}}(\boldsymbol{f}-\boldsymbol{g})\Bigg) \\ \times \int \mathrm{d}\boldsymbol{y}\,\exp\Bigg(&-\frac{1}{2}\boldsymbol{y}^{\mathrm{T}}\left(\frac{\sigma_f^2+\sigma_g^2}{\sigma_f^2\sigma_g^2}\boldsymbol{I} - \boldsymbol{K}_f^{-1}\right)\boldsymbol{y} + \boldsymbol{y}^{\mathrm{T}}\left(\frac{1}{\sigma_f^2}\boldsymbol{f}_{\boldsymbol{X}} + \frac{1}{\sigma_g^2}\boldsymbol{g}_{\boldsymbol{X}}\right)\Bigg) \end{aligned} \tag{6.3.1}$$

which after completing the square for $\boldsymbol{y}$ and marginalising would leave us with

$$\frac{1}{2}\left(\frac{1}{\sigma_f^2}\boldsymbol{f}_{\boldsymbol{X}} + \frac{1}{\sigma_g^2}\boldsymbol{g}_{\boldsymbol{X}}\right)^{\mathrm{T}}\left(\frac{\sigma_f^2+\sigma_g^2}{\sigma_f^2\sigma_g^2}\boldsymbol{I} - \boldsymbol{K}_f^{-1}\right)^{-1}\left(\frac{1}{\sigma_f^2}\boldsymbol{f}_{\boldsymbol{X}} + \frac{1}{\sigma_g^2}\boldsymbol{g}_{\boldsymbol{X}}\right) \tag{6.3.2}$$

in the exponent of the partition function. We cannot however easily deal with this term due to the inverse of a sum of two matrices. We therefore instead linearise the quadratic

$\boldsymbol{y}^{\mathrm{T}}\boldsymbol{K}_f^{-1}\boldsymbol{y}$ term in (6.2.11) first using the identity

$$\exp\left(\frac{1}{2}\boldsymbol{y}^{\mathrm{T}}\boldsymbol{K}_f^{-1}\boldsymbol{y}\right) = \frac{\det^{1/2}\boldsymbol{K}_f}{(2\pi)^{N/2}}\int \mathrm{d}\boldsymbol{h}\, \exp\left(-\frac{1}{2}\boldsymbol{h}^{\mathrm{T}}\boldsymbol{K}_f\boldsymbol{h} + \boldsymbol{y}^{\mathrm{T}}\boldsymbol{h}\right) \tag{6.3.3}$$

This allows us to rewrite (6.3.1) into the form

$$\begin{aligned} Z = \int \mathrm{d}\boldsymbol{f}\,\mathrm{d}\boldsymbol{g}\,\mathrm{d}\boldsymbol{h}\, \exp\Bigg(&-\frac{1}{2}\boldsymbol{f}^{\mathrm{T}}\boldsymbol{C}_f^{-1}\boldsymbol{f} - \frac{1}{2\sigma_f^2}\boldsymbol{f}_{\boldsymbol{X}}{}^{\mathrm{T}}\boldsymbol{f}_{\boldsymbol{X}} - \frac{1}{2\sigma_g^2}\boldsymbol{g}_{\boldsymbol{X}}{}^{\mathrm{T}}\boldsymbol{g}_{\boldsymbol{X}} \\ &- \frac{1}{2}\boldsymbol{g}^{\mathrm{T}}\boldsymbol{C}_g^{-1}\boldsymbol{g} - \frac{\lambda}{2}(\boldsymbol{f}-\boldsymbol{g})^{\mathrm{T}}(\boldsymbol{f}-\boldsymbol{g}) - \frac{1}{2}\boldsymbol{h}^{\mathrm{T}}\boldsymbol{K}_f\boldsymbol{h}\Bigg) \\ &\times \int \mathrm{d}\boldsymbol{y}\, \exp\left(-\frac{\sigma_f^2+\sigma_g^2}{2\sigma_f^2\sigma_g^2}\boldsymbol{y}^{\mathrm{T}}\boldsymbol{y} + \boldsymbol{y}^{\mathrm{T}}\left(\frac{1}{\sigma_f^2}\boldsymbol{f}_{\boldsymbol{X}} + \frac{1}{\sigma_g^2}\boldsymbol{g}_{\boldsymbol{X}} + \boldsymbol{h}\right)\right) \end{aligned} \tag{6.3.4}$$

for $\boldsymbol{h} \in \mathbb{R}^N$ and where we have dropped constant prefactors that will cancel after differentiation with respect to λ. Marginalisation with respect to $\boldsymbol{y}$ then gives

$$\begin{aligned} Z = \int \mathrm{d}\boldsymbol{f}\,\mathrm{d}\boldsymbol{g}\,\mathrm{d}\boldsymbol{h}\, \exp\Bigg(&-\frac{1}{2}\boldsymbol{f}^{\mathrm{T}}\boldsymbol{C}_f^{-1}\boldsymbol{f} - \frac{1}{2\sigma_f^2}\boldsymbol{f}_{\boldsymbol{X}}{}^{\mathrm{T}}\boldsymbol{f}_{\boldsymbol{X}} - \frac{1}{2\sigma_g^2}\boldsymbol{g}_{\boldsymbol{X}}{}^{\mathrm{T}}\boldsymbol{g}_{\boldsymbol{X}} \\ &- \frac{1}{2}\boldsymbol{g}^{\mathrm{T}}\boldsymbol{C}_g^{-1}\boldsymbol{g} - \frac{\lambda}{2}(\boldsymbol{f}-\boldsymbol{g})^{\mathrm{T}}(\boldsymbol{f}-\boldsymbol{g}) - \frac{1}{2}\boldsymbol{h}^{\mathrm{T}}\boldsymbol{K}_f\boldsymbol{h} \\ &+ \frac{\sigma_f^2\sigma_g^2}{2(\sigma_f^2+\sigma_g^2)}\left(\frac{1}{\sigma_f^2}\boldsymbol{f}_{\boldsymbol{X}} + \frac{1}{\sigma_g^2}\boldsymbol{g}_{\boldsymbol{X}} + \boldsymbol{h}\right)^{\mathrm{T}}\left(\frac{1}{\sigma_f^2}\boldsymbol{f}_{\boldsymbol{X}} + \frac{1}{\sigma_g^2}\boldsymbol{g}_{\boldsymbol{X}} + \boldsymbol{h}\right)\Bigg). \end{aligned} \tag{6.3.5}$$

which no longer contains the problematic inverse in (6.3.2).

With the partition function in the form (6.3.5) we are almost in the position to apply the tricks of Chapter 4 and write this as a graphical model by Fourier transforming $\boldsymbol{f}$ and $\boldsymbol{g}$, introducing $2p$ extra variables for each matrix and rewriting sums over examples in terms of γ_i, a count of the number examples seen at vertex i. However, the $N \times N$ $\boldsymbol{K}_f$ matrix still poses a problem to our analysis. This can be resolved by realising that $(\boldsymbol{K}_f)_{\nu,\mu} = C_f(x_\nu, x_\mu) + \sigma_f^2\delta_{\nu,\mu}$. We see then that $\boldsymbol{h}^{\mathrm{T}}\boldsymbol{K}_f\boldsymbol{h}$ is simply equal to $\sum_{i,j}(\sum_\mu \delta_{x_\mu,i}h_\mu)(\boldsymbol{C}_f)_{ij}(\sum_\mu \delta_{x_\mu,j}h_\mu) + \sigma_f^2\sum_\mu h_\mu^2$. Making this substitution into (6.3.5) and introducing the variables, $\boldsymbol{X}_i$, the set of all examples of the teacher at vertex i and,

$h_{\boldsymbol{X}_i} = \sum_\mu \delta_{x_\mu,i} h_\mu$ we see that via a delta enforcement term (6.3.5) reads as

$$Z = \int \mathrm{d}\boldsymbol{f}\,\mathrm{d}\boldsymbol{g}\,\mathrm{d}\boldsymbol{h} \prod_{i=1}^{V} \mathrm{d}h_{\boldsymbol{X}_i} \exp\Bigg(-\frac{1}{2}\boldsymbol{f}^{\mathrm{T}}\boldsymbol{C}_f^{-1}\boldsymbol{f} - \frac{1}{2\sigma_f^2}\boldsymbol{f}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{f}\boldsymbol{X} - \frac{1}{2\sigma_g^2}\boldsymbol{g}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{g}\boldsymbol{X} - \frac{1}{2}\boldsymbol{g}^{\mathrm{T}}\boldsymbol{C}_g^{-1}\boldsymbol{g} - \frac{\lambda}{2}(\boldsymbol{f}-\boldsymbol{g})^{\mathrm{T}}(\boldsymbol{f}-\boldsymbol{g}) - \frac{\sigma_f^2}{2}\boldsymbol{h}^{\mathrm{T}}\boldsymbol{h} - \frac{1}{2}\sum_{i,j} h_{\boldsymbol{X}_i}(\boldsymbol{C}_f)_{ij} h_{\boldsymbol{X}_j} + \frac{\sigma_f^2\sigma_g^2}{2(\sigma_f^2+\sigma_g^2)}\left(\frac{1}{\sigma_f^2}\boldsymbol{f}\boldsymbol{X} + \frac{1}{\sigma_g^2}\boldsymbol{g}\boldsymbol{X} + \boldsymbol{h}\right)^{\mathrm{T}}\left(\frac{1}{\sigma_f^2}\boldsymbol{f}\boldsymbol{X} + \frac{1}{\sigma_g^2}\boldsymbol{g}\boldsymbol{X} + \boldsymbol{h}\right)\Bigg) \times \prod_{i=1}^{V} \delta\left(h_{\boldsymbol{X}_i} - \sum_\mu \delta_{x_\mu,i} h_\mu\right). \quad (6.3.6)$$

After simplification this then becomes

$$Z = \int \mathrm{d}\boldsymbol{f}\,\mathrm{d}\boldsymbol{g} \prod_{i=1}^{V} \mathrm{d}h_{\boldsymbol{X}_i} \exp\Bigg(-\frac{1}{2}\boldsymbol{f}^{\mathrm{T}}\boldsymbol{C}_f^{-1}\boldsymbol{f} - \frac{1}{2}\sum_{i=1}^{V}\left(\frac{\gamma_i}{\sigma_f^2+\sigma_g^2} + \lambda\right) f_i^2 - \frac{1}{2}\sum_{i=1}^{V}\left(\frac{\gamma_i}{\sigma_f^2+\sigma_g^2} + \lambda\right) g_i^2 - \frac{1}{2}\boldsymbol{g}^{\mathrm{T}}\boldsymbol{C}_g^{-1}\boldsymbol{g} - \frac{1}{2}\sum_{i,j} h_{\boldsymbol{X}_i}(\boldsymbol{C}_f)_{ij} h_{\boldsymbol{X}_j} + \frac{\sigma_g^2}{\sigma_f^2+\sigma_g^2}\sum_{i=1}^{V} f_i h_{\boldsymbol{X}_i} + \frac{\sigma_f^2}{\sigma_f^2+\sigma_g^2}\sum_{i=1}^{V} g_i h_{\boldsymbol{X}_i} + \sum_{i=1}^{V}\left(\frac{\gamma_i}{\sigma_f^2+\sigma_g^2} + \lambda\right) f_i g_i\Bigg) \int \mathrm{d}\boldsymbol{h} \exp\left(-\frac{\sigma_f^2}{2}\boldsymbol{h}^{\mathrm{T}}\boldsymbol{h} + \frac{\sigma_f^2\sigma_g^2}{2(\sigma_f^2+\sigma_g^2)}\boldsymbol{h}^{\mathrm{T}}\boldsymbol{h}\right) \prod_{i=1}^{V} \delta\left(h_{\boldsymbol{X}_i} - \sum_\mu \delta_{x_\mu,i} h_\mu\right). \quad (6.3.7)$$

Marginalising over $\boldsymbol{h}$ (see Appendix C) this yields,

$$Z = \int \mathrm{d}\boldsymbol{f}\,\mathrm{d}\boldsymbol{g} \prod_{i=1}^{V} \mathrm{d}h_{\boldsymbol{X}_i} \exp\Bigg(-\frac{1}{2}\boldsymbol{f}^{\mathrm{T}}\boldsymbol{C}_f^{-1}\boldsymbol{f} - \frac{1}{2}\sum_{i=1}^{V}\left(\frac{\gamma_i}{\sigma_f^2+\sigma_g^2} + \lambda\right) f_i^2 - \frac{1}{2}\sum_{i=1}^{V}\left(\frac{\gamma_i}{\sigma_f^2+\sigma_g^2} + \lambda\right) g_i^2 - \frac{1}{2}\boldsymbol{g}^{\mathrm{T}}\boldsymbol{C}_g^{-1}\boldsymbol{g} - \frac{1}{2}\sum_{i,j} h_{\boldsymbol{X}_i}(\boldsymbol{C}_f)_{ij} h_{\boldsymbol{X}_j} + \frac{\sigma_g^2}{\sigma_f^2+\sigma_g^2}\sum_{i=1}^{V} f_i h_{\boldsymbol{X}_i} + \frac{\sigma_f^2}{\sigma_f^2+\sigma_g^2}\sum_{i=1}^{V} g_i h_{\boldsymbol{X}_i} + \sum_{i=1}^{V}\left(\frac{\gamma_i}{\sigma_f^2+\sigma_g^2} + \lambda\right) f_i g_i - \frac{\sigma_f^4}{2(\sigma_f^2+\sigma_g^2)}\sum_{i=1}^{V}\frac{1}{\gamma_i} h_{\boldsymbol{X}_i}^2\Bigg). \quad (6.3.8)$$

where γ_i is set equal to $|\boldsymbol{X}_i|$ and counts the number of examples seen at vertex i in a similar manner to Chapter 4.

With the problematic $\boldsymbol{K}_f$ dealt with, we can now proceed as in the cavity method for the matched case ((4.2.3) through to (4.2.6)). Since the inverse covariance matrices relate

vertices across the entire graph we rewrite these in terms of their Fourier transforms to give,

$$Z = \int \mathrm{d}\boldsymbol{f}\,\mathrm{d}\boldsymbol{\phi}\,\mathrm{d}\boldsymbol{g}\,\mathrm{d}\boldsymbol{\xi} \prod_{i=1}^{V} \mathrm{d}h_{\boldsymbol{X}_i} \exp\Bigg(-\frac{1}{2}\boldsymbol{\phi}^{\mathrm{T}}\boldsymbol{C}_f\boldsymbol{\phi} - \frac{1}{2}\sum_{i=1}^{V}\left(\frac{\gamma_i}{\sigma_f^2+\sigma_g^2}+\lambda\right) f_i^2 - \frac{1}{2}\sum_{i=1}^{V}\left(\frac{\gamma_i}{\sigma_f^2+\sigma_g^2}+\lambda\right) g_i^2 - \frac{1}{2}\boldsymbol{\xi}^{\mathrm{T}}\boldsymbol{C}_g\boldsymbol{\xi} - \frac{1}{2}\sum_{i,j} h_{\boldsymbol{X}_i}(\boldsymbol{C}_f)_{ij}h_{\boldsymbol{X}_j} + \frac{\sigma_g^2}{\sigma_f^2+\sigma_g^2}\sum_{i=1}^{V} f_i h_{\boldsymbol{X}_i} + \frac{\sigma_f^2}{\sigma_f^2+\sigma_g^2}\sum_{i=1}^{V} g_i h_{\boldsymbol{X}_i} + \sum_{i=1}^{V}\left(\frac{\gamma_i}{\sigma_f^2+\sigma_g^2}+\lambda\right) f_i g_i + \mathrm{i}\boldsymbol{\phi}^{\mathrm{T}}\boldsymbol{f} + \mathrm{i}\boldsymbol{\xi}^{\mathrm{T}}\boldsymbol{g} - \frac{\sigma_f^4}{2(\sigma_f^2+\sigma_g^2)}\sum_{i=1}^{V}\frac{1}{\gamma_i}h_{\boldsymbol{X}_i}^2\Bigg). \quad (6.3.9)$$

All that then remains to write this in the form of a graphical model with only nearest neighbour interactions is to rewrite the $\boldsymbol{\phi}^{\mathrm{T}}\boldsymbol{C}_f\boldsymbol{\phi}$, $\boldsymbol{\xi}^{\mathrm{T}}\boldsymbol{C}_g\boldsymbol{\xi}$ and $h_{\boldsymbol{X}_i}(\boldsymbol{C}_f)_{ij}h_{\boldsymbol{X}_j}$ terms, since these relate vertices upto p_f, p_g and p_f away from each other respectively. Once more this is resolved in a similar manner to the matched cavity case ((4.2.4) through to (4.2.5)): By binomially expanding (6.1.1) and introducing additional variables for each of the three terms. After the introduction of the additional variables and writing the Dirac delta enforcement term in its Fourier form we finally arrive at the graphical model

version of (6.2.11) given by (see Appendix C),

$$
\begin{aligned}
Z = \int \mathrm{d}\boldsymbol{f}\,\mathrm{d}\boldsymbol{g}\,\mathrm{d}\boldsymbol{\phi}^0 \prod_{q=1}^{p_f}\left(\mathrm{d}\boldsymbol{\phi}^q\,\mathrm{d}\hat{\boldsymbol{\phi}}^q\right)\,\mathrm{d}\boldsymbol{\xi}^0 \prod_{q=1}^{p_g}\left(\mathrm{d}\boldsymbol{\xi}^q\,\mathrm{d}\hat{\boldsymbol{\xi}}^q\right)\prod_i\left(\mathrm{d}h^0_{\boldsymbol{X}_i}\prod_{q=1}^{p_f}\left(\mathrm{d}h^q_{\boldsymbol{X}_i}\mathrm{d}h^q_{\boldsymbol{X}_i}\right)\right)\\
\prod_{i=1}^{V}\exp\Bigg(-\frac{1}{2}\sum_{q=0}^{p_f}c_{f,q}d_i\phi_i^0\phi_i^q-\frac{1}{2}\left(\frac{\gamma_i}{\sigma_f^2+\sigma_g^2}+\lambda\right)f_i^2\\
-\frac{1}{2}\left(\frac{\gamma_i}{\sigma_f^2+\sigma_g^2}+\lambda\right)g_i^2-\frac{1}{2}\sum_{q=0}^{p_g}c_{g,q}d_i\xi_i^0\xi_i^q-\frac{1}{2}\sum_{q=0}^{p_f}c_{f,q}d_ih^0_{\boldsymbol{X}_i}h^q_{\boldsymbol{X}_i}\\
+\frac{\sigma_g^2 d_i^{1/2}\kappa_f^{1/2}}{\sigma_f^2+\sigma_g^2}f_ih^0_{\boldsymbol{X}_i}+\frac{\sigma_f^2 d_i^{1/2}\kappa_f^{1/2}}{\sigma_f^2+\sigma_g^2}g_ih^0_{\boldsymbol{X}_i}+\left(\frac{\gamma_i}{\sigma_f^2+\sigma_g^2}+\lambda\right)f_ig_i\\
+\mathrm{i}d_i^{1/2}\kappa_f^{1/2}\phi_i^0f_i+\mathrm{i}d_i^{1/2}\kappa_g^{1/2}\xi_i^0g_i-\frac{\sigma_f^4 d_i\kappa_f}{2(\sigma_f^2+\sigma_g^2)\gamma_i}(h^0_{\boldsymbol{X}_i})^2\\
+\mathrm{i}d_i\sum_{q=1}^{p_f}\hat{\phi}_i^q\phi_i^q+\mathrm{i}d_i\sum_{q=1}^{p_g}\hat{\xi}_i^q\xi_i^q+\mathrm{i}d_i\sum_{q=1}^{p_f}\hat{h}^q_{\boldsymbol{X}_i}h^q_{\boldsymbol{X}_i}\Bigg)\\
\times\prod_{(i,j)\in\mathcal{E}}\exp\Bigg(-\mathrm{i}\sum_{q=1}^{p_f}\left[\hat{\phi}_i^q\phi_j^{q-1}+\hat{\phi}_j^q\phi_i^{q-1}\right]-\mathrm{i}\sum_{q=1}^{p_g}\left[\hat{\xi}_i^q\xi_j^{q-1}+\hat{\xi}_j^q\xi_i^{q-1}\right]\\
-\mathrm{i}\sum_{q=1}^{p_f}\left[\hat{h}^q_{\boldsymbol{X}_i}h^{q-1}_{\boldsymbol{X}_j}+\hat{h}^q_{\boldsymbol{X}_j}h^{q-1}_{\boldsymbol{X}_i}\right]\Bigg). \qquad (6.3.10)
\end{aligned}
$$

with $c_{f,q}=\binom{p_f}{q}(1-a_f^{-1})^{p_f-q}(a_f^q)^{-1}$ and $c_{g,q}=\binom{p_g}{q}(1-a_g^{-1})^{p_g-q}(a_g^q)^{-1}$.

For clarity, we define $\mathcal{H}(\ldots_i)$ to be all the terms in the exponent of the first exponential, $\mathcal{J}(\ldots_{ij})$ to be all the terms in the second exponential and relabel $h_{\boldsymbol{X}_i}$ to h_i so that (6.3.10) becomes

$$
\begin{aligned}
Z = \int \mathrm{d}\boldsymbol{f}\,\mathrm{d}\boldsymbol{g}\,\mathrm{d}\boldsymbol{\phi}^0 \prod_{q=1}^{p_f}\left(\mathrm{d}\boldsymbol{\phi}^q\,\mathrm{d}\hat{\boldsymbol{\phi}}^q\right)\mathrm{d}\boldsymbol{\xi}^0 \prod_{q=1}^{p_g}\left(\mathrm{d}\boldsymbol{\xi}^q\,\mathrm{d}\hat{\boldsymbol{\xi}}^q\right)\mathrm{d}\boldsymbol{h}^0\prod_{q=1}^{p_f}\left(\mathrm{d}\boldsymbol{h}^q\,\mathrm{d}\hat{\boldsymbol{h}}^q\right)\\
\times\prod_{i=1}^{V}\exp\Big(-\mathcal{H}(\ldots_i)\Big)\prod_{(i,j)\in\mathcal{E}}\exp\Big(-\mathcal{J}(\ldots_{ij})\Big). \qquad (6.3.11)
\end{aligned}
$$

6.4 Calculating the generalisation error

To see how the generalisation error can be obtained from (6.3.11), we can differentiate $\log Z$ with respect to λ as prescribed by (6.2.10) to get

$$
\epsilon_g = \frac{1}{V}\sum_{i=1}^{V}\left\langle f_i^2+g_i^2-2f_ig_i\right\rangle. \qquad (6.4.1)
$$

Equation (6.4.1) tells us that we will require the variances $\langle f_i^2 \rangle$ and $\langle g_i^2 \rangle$ and the covariance $\langle f_i g_i \rangle$ in order to calculate the generalisation error. Similarly to Chapter 4, we can calculate these using the cavity method: for a large random graph with fixed arbitrary degree sequence the graph is locally tree-like, so that if vertex i were removed the neighbours $j \in \mathcal{N}(i)$ would become approximately independent. The resulting cavity marginals created by removing i, which we will denote $P_j^{(i)}(f_j, g_j, \boldsymbol{h}_j, \hat{\boldsymbol{h}}_j, \boldsymbol{\xi}_j, \hat{\boldsymbol{\xi}}_j, \boldsymbol{\phi}_j, \hat{\boldsymbol{\phi}}_j | \boldsymbol{X})$ where $\boldsymbol{h}_j = (h_j^0, \ldots, h_j^{p_f})^{\mathrm{T}}$ and $\hat{\boldsymbol{h}}_j = (\hat{h}_j^1, \ldots, \hat{h}_j^{p_f})^{\mathrm{T}}$ and other variables are defined similarly, can be calculated iteratively within these subgraphs using the update equations

$$P_j^{(i)}(\ldots_j | \boldsymbol{X}) \propto \exp\Big(- \mathcal{H}(\ldots_j) \Big) \times \int \prod_{k \in \mathcal{N}(j) \setminus i} \mathrm{d}f_k \, \mathrm{d}g_k \, \mathrm{d}\boldsymbol{h}_k \, \mathrm{d}\hat{\boldsymbol{h}}_k \, \mathrm{d}\boldsymbol{\xi}_k \, \mathrm{d}\hat{\boldsymbol{\xi}}_k \, \mathrm{d}\boldsymbol{\phi}_k \, \mathrm{d}\hat{\boldsymbol{\phi}}_k \, \exp\Big(- \mathcal{J}(\ldots_{jk}) \Big) P_k^{(j)}(\ldots_k | \boldsymbol{X}) \quad (6.4.2)$$

This has the same interpretation as the matched case. In terms of the sum-product algorithm, the cavity marginal on the LHS is the message that vertex j sends to the factor in Z for the edge (i, j) (see section 2.3).

We see, by examining the form of $\mathcal{H}$ and $\mathcal{J}$ defined in (6.3.10), that (6.4.2) is solved self-consistently by complex valued Gaussian distributions with zero mean and covariance matrices $\boldsymbol{V}_j^{(i)}$. By performing the Gaussian integrals explicitly, one finds the cavity updates for the covariance matrices are given by

$$\boldsymbol{V}_j^{(i)} = (\boldsymbol{O}_j - \sum_{k \in \mathcal{N}(j) \setminus i} \boldsymbol{X} \boldsymbol{V}_k^{(j)} \boldsymbol{X})^{-1} \quad (6.4.3)$$

where we have defined the $(4p_f + 2p_g + 5) \times (4p_f + 2p_g + 5)$ matrices

$$\boldsymbol{O}_j = \begin{pmatrix} \boldsymbol{O}_{\phi\phi,j} & \boldsymbol{O}_{\xi\phi,j}{}^{\mathrm{T}} & \boldsymbol{O}_{h\phi,j}{}^{\mathrm{T}} \\ \boldsymbol{O}_{\xi\phi,j} & \boldsymbol{O}_{\xi\xi,j} & \boldsymbol{O}_{h\xi,j}{}^{\mathrm{T}} \\ \boldsymbol{O}_{h\phi,j} & \boldsymbol{O}_{h\xi,j} & \boldsymbol{O}_{hh,j} \end{pmatrix} \quad (6.4.4)$$

$$\boldsymbol{X} = \left(\begin{array}{c|cc|c} 0 & 0 \;\cdots\; 0 & & \\ \hline 0 \\ \vdots & \boldsymbol{X}_{\phi\phi} & \mathbf{0} & \mathbf{0} \\ 0 & & & \\ \hline & & 0 & 0 \;\cdots\; 0 \\ \cline{3-4} \mathbf{0} & & 0 & \\ \vdots & \mathbf{0} & & \boldsymbol{X}_{\xi\xi} \\ 0 & & & \\ \hline & & & \\ \mathbf{0} & \mathbf{0} & & \boldsymbol{X}_{hh} \\ & & & \end{array}\right) \tag{6.4.5}$$

with $\boldsymbol{X}_{\phi\phi}$, $\boldsymbol{X}_{\xi\xi}$ and $\boldsymbol{X}_{hh}$ being $(2p_f+1)\times(2p_f+1)$, $(2p_g+1)\times(2p_g+1)$ and $(2p_f+1)\times(2p_f+1)$ matrices defined in a similar manner to (4.2.11) and where the block entries along the diagonal of $\boldsymbol{O}_j$ are given by

$$\boldsymbol{O}_{\phi\phi,j} = \left(\begin{array}{c|ccc|c} \overbrace{}^{f_j} & \overbrace{}^{\boldsymbol{\phi}_j{}^{\mathrm{T}}} & & & \overbrace{}^{\hat{\boldsymbol{\phi}}_j{}^{\mathrm{T}}} & \\ \frac{\gamma_j}{(\sigma_f^2+\sigma_g^2)}+\lambda & -\mathrm{i}d_j^{1/2}\kappa_f^{1/2} & & & & \} f_j \\ \hline -\mathrm{i}d_j^{1/2}\kappa_f^{1/2} & c_{f,0}d_j & \frac{1}{2}c_{f,1}d_j & \cdots\; \frac{1}{2}c_{f,p_f}d_j & & \\ & \frac{1}{2}c_{f,1}d_j & & & -\mathrm{i}d_j & \\ & \vdots & & & \ddots & \} \boldsymbol{\phi}_j \\ & \frac{1}{2}c_{f,p_f}d_j & & & & -\mathrm{i}d_j \\ \hline & & -\mathrm{i}d_j & & & \\ & & & \ddots & & \} \hat{\boldsymbol{\phi}}_j \\ & & & -\mathrm{i}d_j & & \end{array}\right)$$

$$
\boldsymbol{O}_{\xi\xi,j} = \begin{array}{c|ccc|c|}
 & g_j & \boldsymbol{\xi}_j{}^{\mathrm{T}} & & \hat{\boldsymbol{\xi}}_j{}^{\mathrm{T}} & \\
\hline
g_j & \frac{\gamma_j}{(\sigma_f^2+\sigma_g^2)}+\lambda & -\mathrm{i}d_j^{1/2}\kappa_g^{1/2} & & & \\
\hline
\boldsymbol{\xi}_j & -\mathrm{i}d_j^{1/2}\kappa_g^{1/2} & c_{g,0}d_j & \frac{1}{2}c_{g,1}d_j \;\cdots\; \frac{1}{2}c_{g,p_g}d_j & & \\
 & & \frac{1}{2}c_{g,1}d_j & & -\mathrm{i}d_j & \\
 & & \vdots & & \ddots & \\
 & & \frac{1}{2}c_{g,p_g}d_j & & & -\mathrm{i}d_j \\
\hline
\hat{\boldsymbol{\xi}}_j & & & -\mathrm{i}d_j & & \\
 & & & \ddots & & \\
 & & & & -\mathrm{i}d_j &
\end{array}
$$

$$
\boldsymbol{O}_{hh,j} = \begin{array}{c|ccc|c|}
 & \boldsymbol{h}_j{}^{\mathrm{T}} & & & \hat{\boldsymbol{h}}_j{}^{\mathrm{T}} & \\
\hline
\boldsymbol{h}_j & c_{f,0}d_j + \frac{d_j\kappa_f\sigma_f^4}{(\sigma_f^2+\sigma_g^2)\gamma_j} & \frac{1}{2}c_{f,1}d_j \;\cdots\; & \frac{1}{2}c_{f,p_f}d_j & & \\
 & \frac{1}{2}c_{f,1}d_j & & & -\mathrm{i}d_j & \\
 & \vdots & & & \ddots & \\
 & \frac{1}{2}c_{f,p_f}d_j & & & & -\mathrm{i}d_j \\
\hline
\hat{\boldsymbol{h}}_j & & -\mathrm{i}d_j & & & \\
 & & \ddots & & & \\
 & & & -\mathrm{i}d_j & &
\end{array}
$$

where blank areas of these matrices are to be taken to be equal to zero. The remaining off-diagonal block matrices of $\boldsymbol{O}_j$ are also mainly zero and are best summarised using Kronecker delta functions

$$(\boldsymbol{O}_{\xi\phi,j})_{\mu,\nu} = -\delta_{\mu,1}\delta_{\nu,1}\left(\frac{\gamma_i}{\sigma_f^2+\sigma_g^2}+\lambda\right), \qquad \mu = 1,\ldots,2p_g+2 \quad \& \quad \nu = 1,\ldots,2p_f+2$$

$$(\boldsymbol{O}_{h\phi,j})_{\mu,\nu} = -\delta_{\mu,1}\delta_{\nu,1}\left(\frac{\sigma_g^2 d_j^{1/2}\kappa_f^{1/2}}{\sigma_f^2+\sigma_g^2}\right), \qquad \mu = 1,\ldots,2p_f+1 \quad \& \quad \nu = 1,\ldots,2p_f+2$$

$$(\boldsymbol{O}_{h\xi,j})_{\mu,\nu} = -\delta_{\mu,1}\delta_{\nu,1}\left(\frac{\sigma_f^2 d_j^{1/2}\kappa_f^{1/2}}{\sigma_f^2+\sigma_g^2}\right), \qquad \mu = 1,\ldots,2p_f+1 \quad \& \quad \nu = 1,\ldots,2p_g+2.$$

Once more, as we found in the matched case, we have a singularity for γ_j equal to zero. In this case it is in the quadratic h_j^2 term. To resolve this we can use Woodbury's identity

in similar manner to (4.2.15) to calculate the inverse. We also find, in addition to the singularity, that we have a special case for γ_j equal to zero for f_j and g_j, in this situation these terms become purely linear. This also can be dealt with by using the Woodbury identity (see Appendix C).

So far our derivation has assumed a fixed graph. All that now remains is to consider an ensemble of graphs. We consider ensembles characterised by a distribution $p(d)$ of the degrees d_i, with every graph that has the desired degree distribution having equal probability (this encompasses all the graphs in section 2.4). Instead of individual cavity covariances we once more consider a probability distribution $\pi(\boldsymbol{V})$ across all edges in the graph (see section 4.2.2 for more information). The fixed point cavity update equations given in (6.4.3) then correspond to a self-consistency equation for $\pi(\boldsymbol{V})$ given by

$$\pi(\boldsymbol{V}) = \sum_d \frac{p(d)d}{\bar{d}} \left\langle \int \prod_{k=1}^{d-1} \mathrm{d}\boldsymbol{V}_k \, \pi(\boldsymbol{V}_k) \; \delta\left(\boldsymbol{V} - \left(\boldsymbol{O} - \sum_{k=1}^{d-1} \boldsymbol{X}\boldsymbol{V}_k\boldsymbol{X} \right)^{-1} \right) \right\rangle_\gamma \tag{6.4.6}$$

Similarly to the case of the matched generalisation error we have omitted the index j because the vertex label has now become arbitrary. We will assume once more, for simplicity, that the input distribution is uniform, so that the distribution of γ will be Poisson with mean $\nu = V/N$, for V/N fixed as V tends to infinity.

Equation (6.4.6) is solved using population dynamics (see algorithm 4.1). Once the population of covariances for the fixed point equation (6.4.6) has converged, we sample from the population to calculate the generalisation error (see algorithm 4.2) using the graph ensemble version of the generalisation error given by

$$\epsilon(\nu) = \sum_d p(d) \left\langle \int \prod_{k=1}^{d} \mathrm{d}\boldsymbol{V}_k \, \pi(\boldsymbol{V}_k) \; \boldsymbol{e}_{fg}^{\mathrm{T}} \left(\boldsymbol{O} - \sum_{k=1}^{d} \boldsymbol{X}\boldsymbol{V}_k\boldsymbol{X} \right)^{-1} \boldsymbol{e}_{fg} \right\rangle_\gamma \tag{6.4.7}$$

with $(\boldsymbol{e}_{fg})_\mu = \delta_{\mu,1} - \delta_{\mu,2p_f+3}$.

In order to calculate the generalisation error we still require the normalisation constants κ_f and κ_g. These can be calculated in a similar manner to the cavity case (see around (4.2.7)) by performing population dynamics on (6.4.6) with κ_f equal to one, κ_g equal to one and ν equal to zero. Once this population has converged the normalisers can be

calculated via

$$\kappa_f = \sum_d p(d) \int \prod_{k=1}^{d} \mathrm{d}\boldsymbol{V}_k \, \pi(\boldsymbol{V}_k) \; \boldsymbol{e}_f^{\mathrm{T}} \left(\boldsymbol{O} - \sum_{k=1}^{d} \boldsymbol{X} \boldsymbol{V}_k \boldsymbol{X} \right)^{-1} \boldsymbol{e}_f \tag{6.4.8}$$

$$\kappa_g = \sum_d p(d) \int \prod_{k=1}^{d} \mathrm{d}\boldsymbol{V}_k \, \pi(\boldsymbol{V}_k) \; \boldsymbol{e}_g^{\mathrm{T}} \left(\boldsymbol{O} - \sum_{k=1}^{d} \boldsymbol{X} \boldsymbol{V}_k \boldsymbol{X} \right)^{-1} \boldsymbol{e}_g \tag{6.4.9}$$

where $(\boldsymbol{e}_f)_\mu = \delta_{\mu,1}$ and $(\boldsymbol{e}_g)_\mu = \delta_{\mu,2p_f+3}$. This result can be realised from (6.2.2). Running these equations for the case where the teacher and student match (i.e. when the hyperparameters for the student equal those of the teacher, denoted with subscripts f and g respectively) shows that the normalisation constants found with the above equations coincide with the normalisers calculated using the method described just above figure 4.1.

Comparisons between the predicted mismatched generalisation error as a function of ν against numerical simulations are shown in figure 6.1 and figure 6.2 for regular random graphs with fixed degree equal to three, and five hundred vertices. Figure 6.1 shows that the cavity approximations (curves with triangles) accurately trace the numerically calculated curves (curves with filled circles) for a fixed teacher GP prior, and a student GP prior with a range of mismatched noise levels and hyperparameters. We also see, as expected, that the case of matched learning (centre plot red curve) decreases in error fastest than all other cases. The differences between the matched curve (centre plot red line) and the corresponding noise mismatch plots (red lines in top and bottom plots) are very small. This is because the matched curve is at a minimum and the relatively minor change in noise level shown in these plots moves the learning curve only slightly away from this minimum. Somewhat surprisingly, however, we also see that this is the case for the difference between the mismatch curves with a short ranged student and a longer ranged teacher as we change the noise levels of the student (green curves). This result can be explained by noticing that the shorter ranged student may be approximated by a GP with kernel that is equal to zero for all off-diagonal entries and in effect ignores all information from neighbouring vertices, i.e. a GP that learns the function value of each vertex independently of all the other vertices. In this case the learning curve is given by

$$\epsilon_g = \left\langle \frac{\sigma_f^4 + \gamma \sigma_g^2}{(\sigma_f^2 + \gamma)^2} \right\rangle_\gamma \tag{6.4.10}$$

which for large ν will become only weakly dependent on the student's noise values. A

plot of the limiting $p_f/a_f \to 0$ case is shown as a green line in figure 6.1. As we can see as ν becomes large the mismatch curves do indeed collapse onto the curves of the disconnected limiting form.

Figure 6.2 shows the more interesting case of fixed student parameters and with teacher parameters fixed apart from the maximum step size, p_g, which is varied. We can see that once the teacher's kernel becomes significantly shorter ranged than that of the student, a bump above unity appears at about ν equal to 0.2. This bump is caused by underfitting: Since the student is significantly longer ranged it makes incorrect inferences about vertices over long distances which it cannot correct until it sees more examples.

It is worth noting that although the cavity method approximation is accurate in most cases, with the larger covariance matrices needed for the mismatch self consistent equations, the population dynamics algorithm can become unstable. This can be seen in the top plot of figure 6.1 for a student with a maximum number of steps $p_f = 16$, and a low noise level, $\sigma_f^2 = 0.03$.

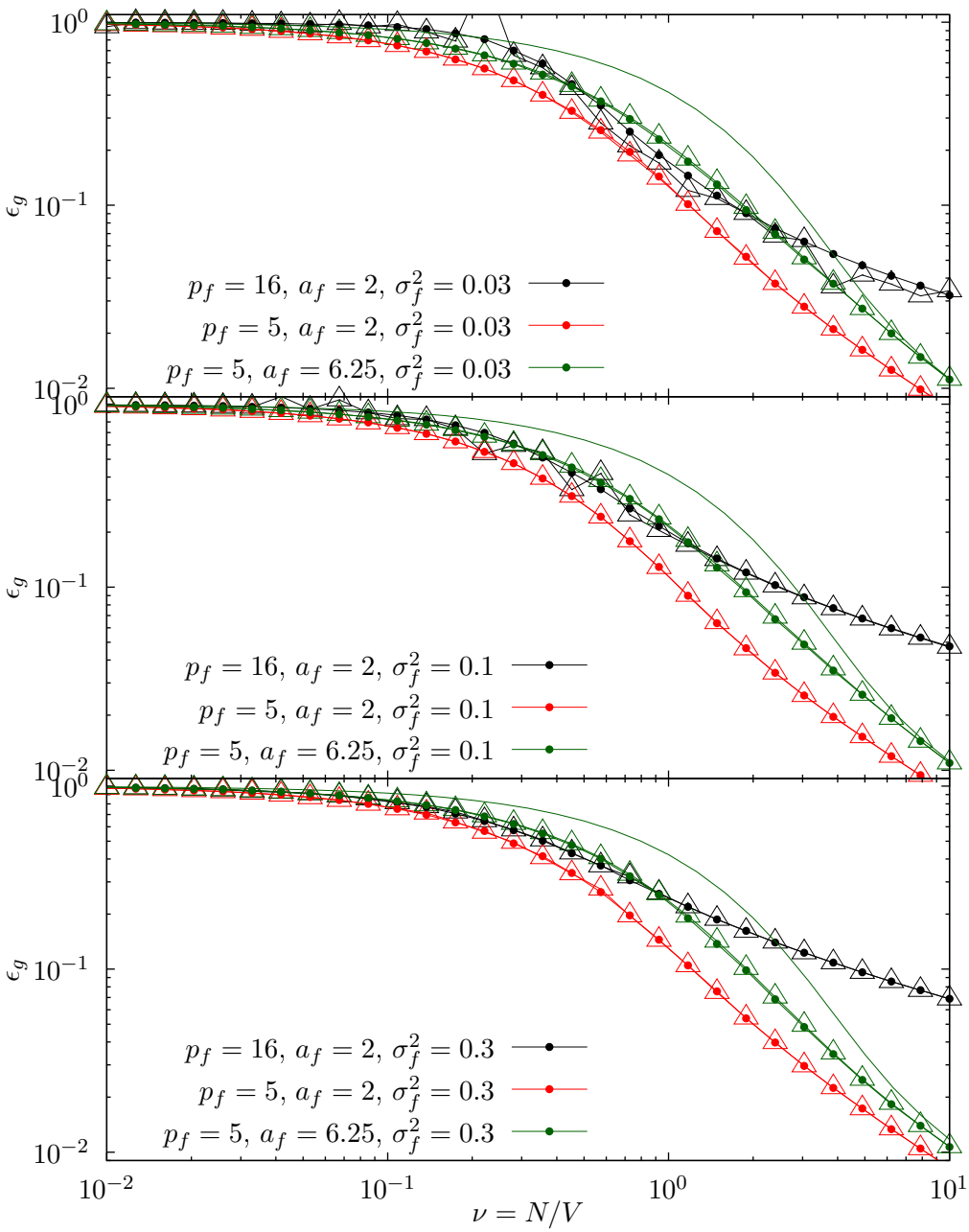


Figure 6.1: Mismatched learning curve numerics compared against cavity derived approximations on a regular graph with degree equal to three and five hundred vertices. Teacher is fixed with $\sigma_g^2 = 0.1$, $p_g = 5$ and $a_g = 2$. Cavity predictions are indicated using triangles and numerics by filled circles. Solid lines show the limiting case of a student with lengthscale equal to zero.

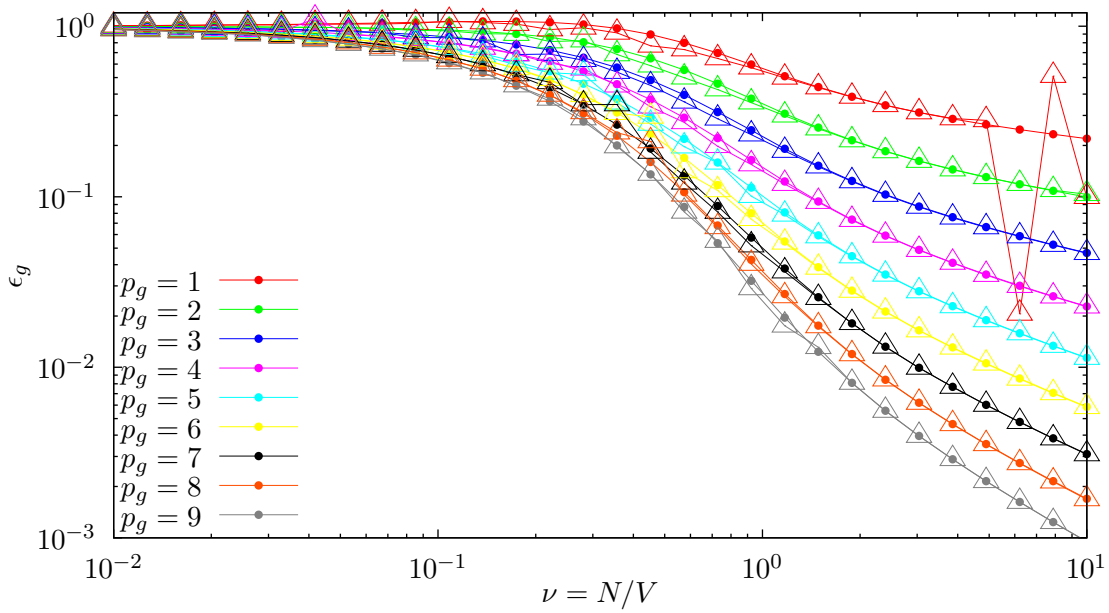


Figure 6.2: Mismatched learning curve numerics compared against cavity derived approximations on a regular graph with degree equal to three and five hundred vertices. Student parameters are $\sigma_f^2 = 0.1$, $p_f = 10$ and $a_f = 2$. Teacher parameters are $\sigma_g^2 = 0.01$, $a_g = 2$ and varying p_g.

6.5 Summary

In this chapter we have shown that we can extend the cavity derivation for the calculation of the generalisation error in a matched scenario where student and teacher GPs have the same hyperparameters to a mismatched scenario where student and teacher hyperparameters are different, for GP priors with globally normalised random walk kernels. We found that, unlike the matched case, we could not make the initial simplification of shifting the posterior so that it had zero mean. Because of this, deriving the relevant generating partition function was technically more difficult.

Our derivation began by trying to calculate the exact form of the joint distribution between the teacher and student functions. We found, however, that to be able to later write the partition function in terms of a graphical model we needed to include an additional variable that encoded information about the student's posterior normalisation. Even with this additional variable in the partition function there were still complicated terms that could not be dealt with as we had in earlier chapters, in particular there was a quadratic term involving the student's Gram matrix that needed to be dealt with. We showed that by writing the Gram matrix explicitly in terms of the student's covariance matrix we could rewrite the partition function in a form similar to the previous case, consisting this time of three separate covariance matrix terms relating vertices in the graph. From here we were able to apply the same tricks as those presented in Chapter 4 to write the partition function in the desired graphical model form.

Once we had the graphical model form we could then solve using the cavity method. The resulting self consistent and generalisation error equations broadly resembled those of Chapter 4, but over a greatly inflated number of variables. In the same way as before we could solve the self-consistency equation using population dynamics and then use this result to calculate the generalisation error as a function of the number of examples or learning curve. Comparing against numerics we found that once more the predictions and simulations were accurate across the whole length of the learning curve so long as the population dynamics converged stably.

Finally we looked at the case of a student with a random walk kernel with a long lengthscale learning from a teacher with a random walk kernel with a short lengthscale.

This comparison was interesting because, as one would expect, the student suffers from overfitting. We showed that for a kernel with a short enough lengthscale the learning curve presented a bump above the initial error, after seeing examples across some of the graph.

CHAPTER 7

CONCLUDING REMARKS

7.1 Summary of Results

In this thesis I have been concerned with the application of GPs for regression on random graphs. I began in Chapter 2 with a literature review and an introduction of the key methods required to understand the results presented in later chapters. In Chapter 3 I then examined the random walk kernel given in (3.1.1) and its use as a kernel in a GP prior. In section 3.1.1 I considered the kernel on a large regular graph and tree. I showed, using the results from Chung and Yau [27] (see also Appendix A), that on the regular tree, which I considered as an approximation of a large regular random graph, the kernel approaches the non-trivial limiting form (3.1.3). Since the regular graph does approach the expected fully correlated limiting form, I inferred that loops must pay an important role in achieving this.

After considering the fully correlated limit of the kernel in section 3.2 I then studied the application of the random walk kernel as a GP prior. I focused in particular on how one could normalise the kernel so as to set the a-priori scale of a function. I showed that the conventional method, which I called global normalisation, where one normalises by dividing by the expected value of the diagonal entries of the unnormalised

kernel, leads to unrealistic probabilistic assumptions. I found in particular that global normalisation leads to large variation in the prior variances of vertices within the graph (see figure 3.3). To remedy this I proposed a different method of normalisation, which I called local normalisation. In this case the entries of the kernel are set to be the Pearson's correlation coefficients of the unnormalised kernel [see for instance 134]. By normalising the covariance function in this way the variation in the scale of the function between vertices was completely eliminated.

Once I had proposed local normalisation as a solution to the variance the prior scale of a GP with a random walk kernel, I then proceeded to examine qualitatively the differences between GP regression with GP priors with random walk kernels with the two normalisations by studying the learning curves. I began in section 3.3 by looking at the case of matched learning curves, where both teacher and student generate functions from the same GP. I showed that local normalisation introduces a fundamental change in the shape of the learning curves (see figure 3.5) by introducing a shoulder when the GP has seen examples on the majority of vertices. I showed that this was caused by correctly normalising disconnected vertices within the graph. Since disconnected vertices cannot make inferences about their value from the rest of the graph they keep their prior variance, and therefore potential error, until an example is seen at that vertex. Disconnected vertices in the case of global normalisation have an atypically small prior variance (see figure 3.3) and so contribute very little to the error of the GP. For local normalisation, however, disconnected vertices have a prior variance fixed to unity. The shoulder introduced into the learning curves of a GP with local normalisation is the result of this increase in their prior variance, which leads to the learning curve being dominated by the disconnected vertices in the region where the GP has seen examples on the majority of the graph (see figure 3.5(a)) but not yet on all of the disconnected vertices.

To further show that local normalisation was the more plausible method of normalising the random walk kernel I then considered the variance of the local Bayes error as a function of the number of examples (see figure 3.7). Plotting these showed that global normalisation began with a high variance in the Bayes error i.e. a large amount of variation between the errors it would make at each vertex. This variance remained

high through to the point at which examples had been seen on most of the graph and subsequently tailed off. Local normalisation, by contrast, began with a low variance in the local Bayes error, peaked at the point where examples had been seen on the majority of the graph and then tailed off at the end. From a probabilistic point of view I argued that local normalisation exhibited more reasonable behaviour; one would expect initially that the GP should be equally uncertain of the function value for any point on the graph, differ greatest when it has seen examples on some parts of the graph and therefore becomes certain of their value and not on others and then fall back to zero as it becomes certain of the function value of all vertices in the graph.

Finally to drive home that these two kernel normalisations led to fundamentally different priors over functions I looked at the learning curves subject to model mismatch (see figure 3.6). In particular I studied the learning curves of a GP with a locally normalised random walk kernel as it tried to learn functions sampled from a GP with a globally normalised kernel. I found that the learning curves were qualitatively fundamentally different with a bump in the generalisation error close to unity at the point where examples were seen on most of the graph. I examined how mismatch changed the way a GP calculates its predictive mean (given in (3.3.1)) and showed it corresponded to a reweighting of 'effective prediction vectors'. I showed by some numerical experiments that the reweighting amounted to assigning additional weight to vectors that corresponded to dangling ends (a series of connected degree two vertices terminating in a degree one vertex) in the graph and that these regions of the graph therefore were the cause of the bump in the learning curve.

With a thorough understanding of how the random walk kernel behaves as a GP prior I then, in a similar vain to Malzahn and Opper [72], Sollich [111], tried to approximate the generalisation error of a GP with a random walk kernel with both global and local normalisation. I began in Chapters 4 and 5 by focussing on the case of where the student and teacher are matched. This meant that I needed to calculate the average posterior variance of the GP (see (4.1.5)). In Chapter 4 I derived a cavity approximation of the learning curve (see section 2.3). I began by first rewriting the generalisation error in terms of a partition function (see (4.2.2)) and then rewrote this in terms of a graphical model which consisted of single vertex and nearest neighbour interactions (see (4.2.6)).

To be able to rewrite the partition function in this form I Fourier transformed the inverse kernel term to eliminate the inverse, and introduced an additional $2p$ variables (see (3.1.1) for the definition of p) to eliminate interactions over distances greater than nearest neighbours. With the generalisation error in this form I then split the remainder of my derivation into two sections depending on the choice of normalisation.

In section 4.2.2 I considered the case of a globally normalised kernel. Using the cavity method I derived a self-consistent equation (see (4.2.13)) and cavity form of the generalisation error (see (4.2.14)). The resulting approximation for the ensemble averaged generalisation error required population dynamics (see algorithm 4.1) to calculate a steady state of a self consistent equation relating the distribution of cavity covariances on the graph to itself. Once the distribution of cavity covariances was found, the generalisation error was simply calculated using Monte Carlo techniques (see algorithm 4.2). Plotting the cavity approximation against numerics and a baseline comparison derived in section 2.2.1 showed that the cavity approximation greatly improved on the baseline, especially in the midsection of the learning curves, becoming visually indistinguishable from numerics for graphs with five hundred vertices.

After deriving the generalisation error approximation for the case of a globally normalised kernel, in section 4.2.3 I considered the technically more difficult problem of approximating the generalisation error of a GP with a locally normalised random walk kernel. Local normalisation requires knowledge about local structure of the specific graph instance; this means that unlike the global case, the normaliser must be calculated in parallel with the cavity covariances. To include this requirement I began by introducing an additional step in the cavity calculation to calculate the local normalising factors (see (4.2.3)). This step could be solved in a similar manner to the previous globally normalised case and required the introduction of an additional set of auxiliary variables that calculated the normalisation constants for the partition function (see (4.2.23)). Once I had introduced these auxiliary terms I could again use the graphical model form of the partition function which could be simply solved on a specific graph instance using BP. Extending this solution to the graph ensemble case, however, proved to be non-trivial. It required an additional approximation, namely, dropping the requirement of consistency between incoming auxiliary covariance matrices. After the approximation I once

more had a self-consistent equation over the distribution of the auxiliary and normal covariance matrices that could be solved using population dynamics. Once a steady state of the self-consistent equation was found the generalisation error was again simply a matter of running Monte Carlo integration over the distribution of cavity covariances. Plotting the results of this approximation of the generalisation error as a function of the number of examples, or learning curve, against numerics again showed good agreement along the length of the curve, surprisingly so given the additional approximation with respect to consistency (see figure 4.4).

To gain a deeper understanding of the approximations required to derive the cavity approximation of the locally normalised kernel, in Chapter 5, I then derived the approximations of the generalisation error for both globally and locally normalised kernels using the replica method, assuming replica symmetry (see section 2.2). Once more this was done by rewriting the generalisation error in terms of the partition function given in (4.2.6). Derivation of the GP with a globally normalised random walk kernel then loosely followed the derivation of the eigenvalues of a matrix given in Kühn et al. [66]. The derivation of the globally normalised case served two purposes: it gave a summary of the method I would employ in approximating the locally normalised case, and served as a way of interpreting the locally normalised equations. To calculate the approximation of the generalisation error for the globally normalised kernel I rewrote the graphical model form of the partition function in terms of an effective single site form. Once in this form I could then invoke a large graph saddle point approximation that gave a set of simultaneous equations to be solved (see (5.1.30) and (5.1.31)). I found that one solution to this set of equations was of quadratic form and resulted, as expected, in the same covariance update equations as obtained using the cavity approach (see (5.1.37)).

The derivation of the generalisation error for the locally normalised kernel was substantially more complicated since, unlike the cavity derivation, I needed to include the constraint on the normalising factors directly into the partition function. This was achieved by using a delta enforcement term (see (5.2.1)), which then broke the graphical model form required to apply the techniques used to calculate the global generalisation error. To resolve this I showed that through a series of steps the partition function could once more be written in the form of a graphical model. This required two additional sets

of replicas, one to rewrite a term in the denominator into the numerator, and another to replace an average by an empirical average. With the introduction of these additional replicas I was able once more to rewrite this into an equation consisting of single vertex and nearest neighbour interactions only, by using similar techniques to those employed in creating the original graphical model form (see (5.2.8)).

Once I had the partition function in this form I could begin approximating the generalisation error by following a similar method to the globally normalised case; by deriving an effective single site formulation and invoking a large vertex saddle point approximation. Eventually however, I also required a large replica limit of the additional replicas introduced by the normalising constraints. This large replica limit needed to be taken internally of the original large V saddle point equations. The replica limit was calculated using the saddle point method (see Appendix B.1.2) and allowed me to approximate a complicated integral over the normalising factors by its minimum along the integration path, simplifying the simultaneous equations generated by the large graph saddle point (see (5.2.31) and (5.2.32)) and finally resulting in the self consistency equation (5.2.17).

With the large replica limit taken I could state the form of one solution to the self consistency equations. This form was quadratic with one auxiliary independent covariance matrix relating auxiliary variables and another covariance function that was dependent on the auxiliary variables relating the remaining variables (see (5.2.34)). Substituting this form into the self consistency equation then gave an equation which, in the global case, I would be able to solve with population dynamics (see (5.2.41)). For the locally normalised case however, the complex interaction between the two covariance matrices meant that population dynamics could not be applied. I noted, by working backwards from the generalisation error (see (5.2.43)), that in fact I only required the covariance matrix function of the remaining variables evaluated at the full marginal of the auxiliary variables. By taking advantage of self averaging I was then able to simplify the matrix function's self consistency equations resulting in a matrix consistency equation that required a counterflow of information about the auxiliary variables from the receiving vertex (see figure 5.1). Although this simplified matters, the self consistency equation still presented difficulties because it included a consistency term constraining the auxiliary covariances from the incoming vertices (see (5.2.49)). To resolve this I made an

approximation, and ignored the consistency requirement. This approximation resulted in exactly the same equations that were found using the cavity method in Chapter 4 (see (5.2.51)). That this gave accurate results, of a level of accuracy comparable to the results from the global normalisation, was surprising and leads me to believe that the approximation is possibly exact. However, I was unable to find an argument to confirm this.

Finally in Chapter 6 I generalised the cavity approximation of the generalisation error with a globally normalised kernel to the case of model mismatch. Since I could not use the simplifications in the generalisation error that occur when teacher and student are matched (see section 2.1.4.2), the initial steps in rewriting the generalisation error in terms of a graphical model proved to be a more challenging task. This problem was tackled in section 6.2 and section 6.3. I began by rewriting the probability distribution over the variables as a single joint distribution that I needed to average (see (6.2.8)). Once I had done this I was able to rewrite the generalisation error in the form of a partition function (see (6.2.11)). It then remained to rewrite the partition function in terms of single vertex and nearest neighbour interactions. This was completed by first linearising the output terms in the exponent of the partition function, marginalising over the now linear output terms (see (6.3.5)), using three applications of the Fourier transform trick to eliminate inverses and introducing $2p_f$, $2p_g$ and $2p_f$ additional variables (see (6.1.1) for the definition). With the partition function in the form of a graphical model I could apply the cavity method in direct analogy to Chapter 4 to derive the self consistent equation given in (6.4.6) and a generalisation error approximation given in (6.2.10).

To end Chapter 6 I compared the prediction given in (6.2.10) to numerically calculated learning curves for a range of mismatched kernel parameters (see figure 6.1 and figure 6.2). I found that in all cases the cavity prediction was accurate along the whole length of the curve, although the population dynamics became numerically unstable for larger matrix sizes. Plotting the more extreme cases of a student with a small lengthscale and a teacher with a much longer lengthscale showed that the GP suffered from overfitting. This overfitting resulted in an initial decrease in the generalisation error that needed to be 'un-learnt' after seeing a larger number of examples. This gave rise

to a bump in the learning curve as seen in figure 6.2.

7.2 Further Work

The work I have presented in this thesis can be extended in a number of ways: In Chapter 3 I have only considered two normalisations of the learning curve, while in fact there is a range of choices of normalisation that could be chosen such that it fits into the form given in (4.1.2). These correspond in some way to changing the probabilities of random walks on the graph. Of particular interest would be to consider a kernel that uses maximal entropy random walks that favour vertices based on their 'centrality' in the graph and thus weight neighbouring data differently.

The random walks used in the kernel defined in (3.1.1) weight probability of passing to a neighbouring vertex equally amongst all the neighbouring vertices; this can be viewed as selecting a walk that maximises entropy locally at each step. Maximum entropy random walks, however, aim to maximise the entropy over the set of all walks of a fixed length starting from a vertex, i.e. they aim to select the random walks of a fixed length k starting at a vertex i uniformly from the set of all such walks, thus the maximum entropy random walk can be thought of as the most random random walk. The maximum entropy random walk has some interesting properties that make it distinct from the random walk used in this thesis. Of particular interest is that it has been shown that the maximum entropy random walk gives a different measure of the centrality or importance of a vertex and mixes faster [34]. This, in terms of a GP kernel, should reweight the relative importance of each vertex in the graph based on the number of different paths that pass through the vertex and may result in more realistic assumptions about the interactions between vertices in the graph. To use this as a random walk kernel one requires an extension of the normalised graph Laplacian, this can be obtained by left and right multiplying the maximum entropy random walk probability matrix by the square root of diagonal matrices consisting of the steady state probabilities of the walk [85].

Although creating the kernel should be simple enough, the theoretical analysis of the learning curves for the maximum entropy random walk kernel could prove challenging. Maximum entropy random walks require knowledge of the complete graph to be cal-

culated, while the cavity and replica methods presented here will only work with local knowledge. To resolve this the work of Sinatra et al. [108] could help; in this paper the authors constructed approximately maximum random walk kernels using only properties of the current vertex degree and degree-degree correlations in the graph.

In the same vein as maximum entropy random walks further improvements could be made by extending the comparison between the global and local normalisations. There are two main directions this could take, firstly one could apply a more rigorous approach to arguing in favour of either normalisation by comparing how each kernel performs on real world data. To do this one would first need to apply maximum marginal likelihood to select the best kernel parameters for each [96] and then compare their performance (or resulting marginal likelihoods) against one another. The second way one could extend this section is to use the analysis of the mismatched learning curves in Chapter 6 to calculate theoretical approximations of the learning curves using the cavity method. In theory this would simply require the steps needed to calculate learning curve for local normalisation in the matched case applied to either the teacher or student variables in the mismatch derivation so that the additional auxiliary variables calculate the local normalisation constants required for the teacher or student's kernel. Due to the already unstable numerics that we saw in the mismatch learning curves, however, the convergence times of the population dynamics could prove to be infeasible.

In chapters 4 and 5 I only considered the case of a graph ensembles that can be described by a degree distribution and where degrees are sampled i.i.d. from this distribution. In reality, as I discussed in section 2.4, real world graphs contain a far richer set of properties than this. One could extend the results of these chapters by applying the work of Kühn and van Mourik [65], Pérez-Vicente and Coolen [94], Rogers et al. [98] where graphs with a modular structure are considered. These have degree correlations and generalised random degrees, where the n-th generalised degree is a sum of the neighbours' $(n-1)$-th generalised degree and where the first generalised degree is defined to be the conventional degree used in this thesis. In particular, the results I have generated so far should be simple to generalise to community structured graphs. These graphs have a 'superstructure' in the form of a tree-like graph which includes graph ensembles discussed in this thesis, but then an added layer of detail is included so that instead

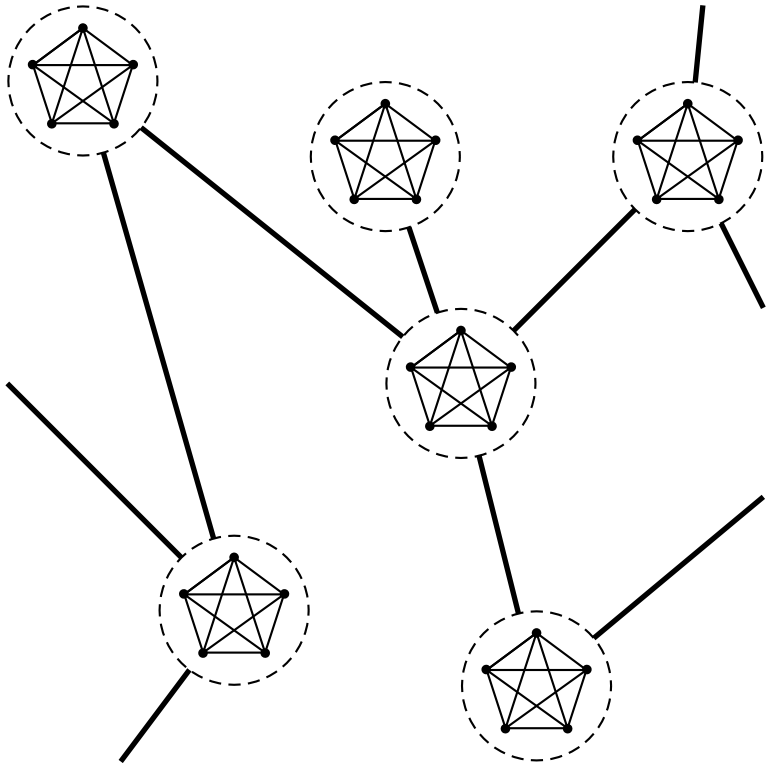


Figure 7.1: A diagramtic view of the community random graph ensemble

of single vertices interacting along connecting edges there are 'communities' interacting with each other using groups of edges between the two communities. The superstructure selects which communities interact with each other (see figure 7.1).

The probability distribution for a random graph ensemble with a community graph structure is given by

$$P(\boldsymbol{A}|\{d_i\}) = \prod_{i<j}\left(\frac{\bar{d}}{V}a_{ij}\mu(\boldsymbol{A}^{ij}_{\text{inter}}) + (1-\frac{\bar{d}}{V})(1-a_{ij})\right)\prod_i \rho(\boldsymbol{A}_{\text{intra}})\delta_{d_i,\sum_j a_{ij}} \qquad (7.2.1)$$

where ρ is a distribution of matrices representing the distribution over local community structure at each 'super vertex', μ is distribution of edges connecting vertices between communities and $\{d_i\}$ is some random sequence of degrees generated i.i.d. from a degree distribution $p(d)$. This will create a graph so that the superstructure is treelike, allowing us to apply the techniques of Chapter 4 and Chapter 5.

With a graph sampled from such a community ensemble outlined above, the cavity method approach would simply now pass covariances concerning the whole community of vertices; local vertex contributions denoted by $\boldsymbol{O}$ in this thesis (see (4.2.10)) would

encode the entire community's local data including local interactions and finally interactions between communities encoded by the matrices $\boldsymbol{X}$ would be of the same form as (4.2.11), but for each entry in (4.2.11) there would now be a matrix representing the interconnections between the communities. Generalising the covariance matrices in this way and following a similar method to that in Chapter 4 then would give a self consistent equation similar to (4.2.13), but instead of sampling a single vertex, one would sample a community with each Monte Carlo step and calculate the generalisation error for the whole community. Preliminary results have shown that this approximation of the generalisation error shows promise and is shown in figure 7.2.

Figure 7.2 shows the matched numerical learning curves of a GP (indicated by filled circles) learning on a community graph ensemble, compared to the cavity prediction (indicated with triangles) calculated using the method outlined above. These results are for a community ensemble, consisting of a super structure given by an Erdős-Rényi random graph, with communities of four vertices all connected to each other and interconnections given by randomly connecting one vertex between two communities.

Another avenue one could take to extend the classes of graph ensembles for which the results presented in this thesis are applicable is to consider weighted graphs. In this situation edges also have a given numerical value that indicates the amount of interaction between two vertices. The random walk kernel would then be defined using the more general normalised Laplacian given in Appendix A. Ensembles of this type can be defined by introducing a second continuous random variable to the ensembles already discussed that generates i.i.d. weights for the edges when an edge is present [17].

Further interesting areas which can be expanded in this work are found in Chapter 6, the most promising of which is to consider the adaptation of the student parameters as the number of examples increases along the lines of [113, 114]. In these papers the authors don't consider the case where only the hyperparameters between student and teacher are mismatched but instead consider a teacher GP with a completely different kernel. A simple and more restrictive way to implement this in the framework of the cavity method above would be to consider a teacher GP with the random walk kernel and fixed hyperparameters, and a student GP with the random walk kernel with one hyperparameter fixed and set different from the teacher's and use maximum marginal likelihood to

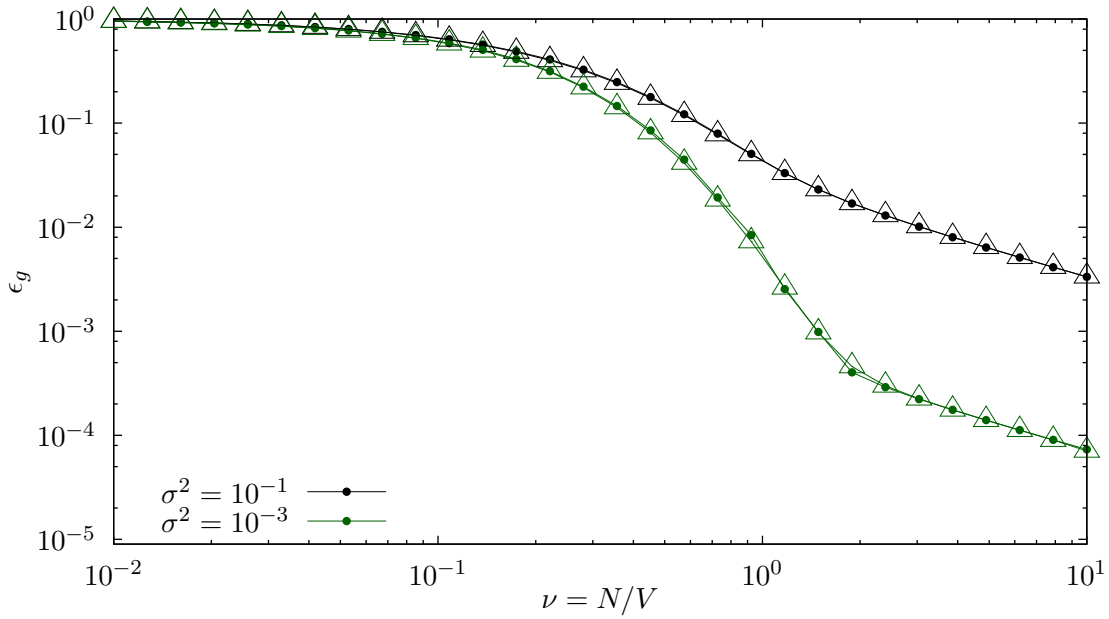


Figure 7.2: A comparison between numerically calculated learning curves and the cavity prediction for a community random graph with five hundred communities each consisting of four vertices all connected to each other. Interactions between communities are generated by randomly connecting a vertex from each community so long as the superstructure given by an Erdős-Rényi graph with average degree three indicates there should be an edge. Kernel parameters are $a = 2$ and $p = 10$.

calculate the remaining hyperparameters. The marginal likelihood is computed using the methods presented in Rasmussen and Williams [96]. This could be extended further by considering a teacher GP with graph kernel that can be accurately approximated by a truncated sum of powers of the normalised Laplacian. In this situation the above cavity analysis for model mismatch could still be applied. The 'mismatch with adaptive hyperparameters' scenario represents a more realistic situation if one wanted to apply GP regression to a real world problem. Applying this to the equations given in Chapter 6, however, could potentially prove tricky due to the numerical stability of the population dynamics. One might expect that so long as the mismatch is with regards to the maximum number of steps of the random walk kernels of the teacher and the student so that the main source of the instability is controlled, numerical stability may be sufficient to get reliable results.

Perhaps more interesting still in the area of model mismatch is to consider the case of graph mismatch. In this case one could consider looking at randomly removing and adding some edges to the graph that the student GP is learning on. This would be a more accurate representation of real world models where, as in the case of biology, the true underlying network is unknown because it has only been inferred from experimental data [99], or in the case of the internet based graphs, where the graph structure is a snapshot that slowly becomes less accurate or worse consists of regions of snapshots which have been updated at different times based on when a web crawler last visited a given region of the internet [14]. One might be able to include graph mismatch into the into the replica framework used in this thesis by considering the methods presented in Annibale and Coolen [7] where the authors considered the problem of modelling the sampling of a real network using experiments. They did this by introducing Bernoulli random variables that indicated changes in the observed network over that of the true underlying network. The authors considered three types of random mistakes vertex undersampling, edge undersampling and edge oversampling and showed that, using methods from statistical mechanics, that they were able to describe the topological properties of the resulting network. For the case of predicting the learning curves subject to model mismatch the simplest method along these lines would be to introduce two additional random Bernoulli variables σ_{ij}^{a} with probability of success $O(1/V)$ and σ_{ij}^{r} with probability of success $O(1)$, for the addition and removal of an edge in the graph respectively.

With these additional random variables the interactions between vertices for the student kernel would then be given by $(1 - \sigma_{ij}^{\mathrm{r}})a_{ij} + (1 - a_{ij})\sigma_{ij}^{\mathrm{a}}$. By varying the probability of success of σ_{ij}^{a} and σ_{ij}^{r} one would be able to change the amount of graph mismatch.

Appendices

APPENDIX A

GRAPH COVERINGS AND THE RANDOM WALK KERNEL

This appendix details the adjustments to Chung and Yau [27] needed to derive the limiting form of the random walk kernel and the large p scaling of the eigenvalue learning curve predictions.

A.1 Graph coverings

Chung and Yau [27] were concerned with the relationships between two non-simple weighted graphs when one of the graphs *covers* the other. A weighted non-simple graph $G(\mathcal{V}, \mathcal{E})$ has the property that edges are allowed to start and finish at the same vertex, and also each edge has a weight associated with it according to some function $w : \mathcal{V} \times \mathcal{V} \to \mathbb{R}$. A graph covering is defined as,

Definition A.1.1 (Graph covering [27])**.** A weighted non-simple graph $\tilde{G}(\tilde{V}, \tilde{E})$ with weight function $\tilde{w} : \tilde{\mathcal{V}} \times \tilde{\mathcal{V}} \to \mathbb{R}$ is said to cover a graph $G(\mathcal{V}, \mathcal{E})$ with weight function $w : \mathcal{V} \times \mathcal{V} \to \mathbb{R}$ if there is a mapping $\pi : \tilde{\mathcal{V}} \to \mathcal{V}$ with the following properties

1. There exists a $m \in \mathbb{R}^+ \cup \{\infty\}$ such that for $u, v \in \mathcal{V}$

$$\sum_{\substack{x \in \pi^{-1}(u) \\ y \in \pi^{-1}(v)}} \tilde{w}(x, y) = m w(u, v) \tag{A.1.1}$$

2. For all $x, y \in \mathcal{V}$ with $\pi(x) = \pi(y)$ and $v \in \mathcal{V}$

$$\sum_{z \in \pi^{-1}(v)} \tilde{w}(z, x) = \sum_{z \in \pi^{-1}(v)} \tilde{w}(z, y). \tag{A.1.2}$$

The authors studied the relationship between the diffusion kernel of $\tilde{G}$ and the diffusion kernel of G [27, Lemma 4] with a generalised version of (2.1.25) given by

$$\boldsymbol{L}_{uv} = \begin{cases} 1 - \dfrac{w(v,v)}{d_v} & \text{if } u = v \text{ and } d_v \neq 0 \\ -\dfrac{w(u,v)}{\sqrt{d_u d_v}} & \text{if } u \text{ and } v \text{ are adjacent} \\ 0 & \text{otherwise,} \end{cases} \tag{A.1.3}$$

and where the 'degree' is given by $d_u = \sum_v w(u, v)$. Chung and Yau [27] showed that for a diffusion kernel, C, on a graph G and a diffusion kernel, $\tilde{C}$, on $\tilde{G}$ where $\tilde{G}$ is a covering of G that

$$\sum_{x \in \pi^{-1}(u)} \sum_{y \in \pi^{-1}(v)} \tilde{C}_{xy} = \sqrt{|\pi^{-1}(u)||\pi^{-1}(v)|} C_{uv} \tag{A.1.4}$$

To derive the limiting form of the random walk kernel we generalise this result to relate random walk kernels of $\tilde{G}$ and G, this results in the following lemma based on [27, Lemma 4].

Lemma A.1.1. Suppose $\tilde{G}$ is a covering of G. Let $\tilde{C}$ and C represent the random walk kernels on each graph respectively. Then

$$\sum_{x \in \pi^{-1}(u)} \sum_{y \in \pi^{-1}(v)} \tilde{C}_{xy} = \sqrt{|\pi^{-1}(u)||\pi^{-1}(v)|} C_{uv} \tag{A.1.5}$$

Proof. We begin by using the binomial expansion result (3.1.1) to give

$$C_{uv} = \sum_{q=0}^{p} \binom{p}{q} (a^{-1})^q ((-\boldsymbol{L})^q)_{uv} \tag{A.1.6}$$

In line with our earlier discussion of the interpretation of the random walk kernel we see that $((-\boldsymbol{L})^q)_{uv}$ is simply a weighted sum of all lazy walks of length q joining u to v where each walk $u_0, u_1, \ldots, u_q$ is given weight equal

$$\prod_{i=1}^{q}\left(\frac{w(u_{i-1},u_i)}{\sqrt{d_{u_{i-1}}d_{u_i}}}-\delta_{u_{i-1}u_i}\right). \tag{A.1.7}$$

Here as in the main text we have defined a lazy walker as one that is also allowed to stay at its current vertex in a time step.

We will denote the weighted sums of random walks in the graphs G and $\tilde{G}$ by $S_q(u,v)$ and $\tilde{S}_q(x,y)$ respectively. To prove (A.1.5) we will show that the sum of $\tilde{S}_q(x,y)$ between the vertices x in the pre-image of πu to vertices y in the pre-image of πv is equal to $S_q(u,v)$ from u to v multiplied by the factor on the RHS of (A.1.5). To do so we need the following relationship which follows directly from substituting property 1. into 2.

$$|\pi^{-1}(v)|\sum_{x\in\pi^{-1}(u)}\tilde{w}(x,z)=mw(u,v)\qquad\forall z\in\pi^{-1}(v). \tag{A.1.8}$$

We will use this result after summing both sides over u, in which case the equation reduces to $|\pi^{-1}(v)|d_z=md_v$.

Using induction and the above equation we will now prove (A.1.5). We will do this by focussing on the relationship between $S_q(u,v)$ and $\tilde{S}_q(x,y)$. We wish to show that

$$\sum_{x\in\pi^{-1}(u)}\sum_{y\in\pi^{-1}(v)}\tilde{S}_q(x,y)=\sqrt{|\pi^{-1}(u)||\pi^{-1}(v)|}S_q(u,v) \tag{A.1.9}$$

for all $q\geq 1$.

We begin by looking at $q=1$, where we have for $u\neq v$

$$\begin{aligned}
&\sum_{x\in\pi^{-1}(u)}\sum_{y\in\pi^{-1}(v)}\tilde{S}_1(x,y)\\
&\quad=\sum_{x\in\pi^{-1}(u)}\sum_{y\in\pi^{-1}(v)}\frac{\tilde{w}(x,y)}{\sqrt{d_xd_y}}=\sum_{x\in\pi^{-1}(u)}\sum_{y\in\pi^{-1}(v)}\frac{\tilde{w}(x,y)}{\sqrt{\sum_z\tilde{w}(x,z)\sum_{z'}\tilde{w}(y,z')}}\\
&\quad=\sum_{x\in\pi^{-1}(u)}\sum_{y\in\pi^{-1}(v)}\frac{\tilde{w}(x,y)}{\sqrt{\sum_{u'\in\mathcal{V}}\sum_{z\in\pi^{-1}(u')}\tilde{w}(x,z)\sum_{v'\in\mathcal{V}}\sum_{z'\in\pi^{-1}(v')}\tilde{w}(y,z')}}\\
&\quad=\sum_{x\in\pi^{-1}(u)}\sum_{y\in\pi^{-1}(v)}\frac{\tilde{w}(x,y)}{\sqrt{md_u/|\pi^{-1}(u)|md_v/|\pi^{-1}(v)|}}\\
&\quad=\sqrt{|\pi^{-1}(u)||\pi^{-1}(v)|}\frac{w(u,v)}{\sqrt{d_ud_v}}=\sqrt{|\pi^{-1}(u)||\pi^{-1}(v)|}S_1(u,v)
\end{aligned} \tag{A.1.10}$$

and for $u = v$

$$\sum_{x\in\pi^{-1}(u)}\sum_{y\in\pi^{-1}(u)}\tilde{S}_1(x,y) = \sum_{x\in\pi^{-1}(u)}\sum_{y\in\pi^{-1}(u)}\left(\frac{\tilde{w}(x,y)}{\sqrt{d_x d_y}} - \delta_{x,y}\right)$$
$$= \sqrt{|\pi^{-1}(u)||\pi^{-1}(u)|}\frac{w(u,u)}{d_u} - |\pi^{-1}(u)| = |\pi^{-1}(u)|S_1(u,u) \quad \text{(A.1.11)}$$

so we see (A.1.9) holds true for this case.

We will next assume that the result (A.1.9) holds for $q-1$ and show that in this case it holds true for q. Let $p_{q-1} = (u, u_1, \ldots, u_{q-1})$ be a walk in G of length $q-1$ that starts at u, let $\tilde{p}_{q-1} = (x, x_1, \ldots, x_{q-1})$ be a walk in $\tilde{G}$ and finally let P and $\tilde{P}$ be the set of all walks that start at u and end at u_{q-1}, and the set of all walks that are mapped onto P by π respectively. Assuming that (A.1.9) holds true for $q-1$ we see that

$$\sum_{x\in\pi^{-1}(u)}\sum_{x_{q-1}\in\pi^{-1}(u_{q-1})}\tilde{S}_{q-1}(x,x_{q-1}) = \sum_{\tilde{p}_{q-1}\in\tilde{P}}\tilde{w}(\tilde{p}_{q-1})$$
$$= \sqrt{|\pi^{-1}(u)||\pi^{-1}(u_{q-1})|}\sum_{p_{q-1}\in P}w(p_{q-1}) \quad \text{(A.1.12)}$$
$$= \sqrt{|\pi^{-1}(u)||\pi^{-1}(u_{q-1})|}S_{q-1}(u,u_{q-1})$$

where we have denoted the weight of a walk p_{q-1} using $w(p_{q-1})$. We wish to extend p_{q-1} to $p_q = u, u_1, \ldots, u_{q-1}, v$ so that it corresponds to a walk from u to v in G. We see that the weight of any path in $\tilde{G}$ that is mapped to p_q can be written as a path p_{q-1} with an additional jump to $y \in \pi^{-1}(v)$ with weight

$$w((\tilde{p}_{q-1}, y)) = \tilde{w}(\tilde{p}_{q-1})\left(\frac{\tilde{w}(x_{q-1},y)}{\sqrt{d_{x_{q-1}}d_y}} - \delta_{x_{q-1},y}\right) \quad \text{(A.1.13)}$$

By summing over all walks that map to p_{q-1} and all additional final jumps that end in $y \in \pi^{-1}(v)$ we will have summed over all walks in $\tilde{G}$ that are mapped to a walk from u to v in G.

If we keep the last but one point u_{q-1} fixed for now then we see that (note that x_{q-1} is not fixed since $\tilde{P}$ is defined as being all paths mapped onto P by π and so x_{q-1} can

take any vertex in $\pi^{-1}(u_{q-1})$),

$$\begin{aligned}
&\sum_{\tilde{p}_{q-1}\in\tilde{P}} \tilde{w}(\tilde{p}_{q-1}) \sum_{y\in\pi^{-1}(v)} \left(\frac{\tilde{w}(x_{q-1},y)}{\sqrt{d_{x_{q-1}} d_y}} - \delta_{x_{q-1},y} \right) \\
&\quad = \sum_{\tilde{p}_{q-1}\in\tilde{P}} \tilde{w}(\tilde{p}_{q-1}) \left(\frac{mw(u_{q-1},v)/|\pi^{-1}(u_{q-1})|}{\sqrt{md_{u_{q-1}} md_v/(|\pi^{-1}(u_{q-1})||\pi^{-1}(v)|)}} - \delta_{u_{q-1},v} \right) \\
&\quad = \sum_{\tilde{p}_{q-1}\in\tilde{P}} \tilde{w}(\tilde{p}_{q-1}) \left(\frac{w(u_{q-1},v)}{\sqrt{d_{u_{q-1}} d_v}} \sqrt{\frac{|\pi^{-1}(v)|}{|\pi^{-1}(u_{q-1})|}} - \delta_{u_{q-1},v} \right) \\
&\quad = \sqrt{|\pi^{-1}(u)||\pi^{-1}(u_{q-1})|} \sum_{p_{q-1}\in P} w(p_{q-1}) \left(\frac{w(u_{q-1},v)}{\sqrt{d_{u_{q-1}} d_v}} \sqrt{\frac{|\pi^{-1}(v)|}{|\pi^{-1}(u_{q-1})|}} - \delta_{u_{q-1},v} \right) \\
&\quad = \sqrt{|\pi^{-1}(u)||\pi^{-1}(v)|} \sum_{p_{q-1}\in P} w(p_{q-1}) \left(\frac{w(u_{q-1},v)}{\sqrt{d_{u_{q-1}} d_v}} - \delta_{u_{q-1},v} \right) \\
&\quad = \sqrt{|\pi^{-1}(u)||\pi^{-1}(v)|} \sum_{p_{q-1}\in P} w((p_{q-1},v))
\end{aligned} \tag{A.1.14}$$

Summing both sides over $u_{q-1} \in \mathcal{V}$ finally gives

$$\sum_{x\in\pi^{-1}(u)} \sum_{y\in\pi^{-1}(v)} \tilde{S}_q(x,y) = \sqrt{|\pi^{-1}(u)||\pi^{-1}(v)|} S_q(u,v) \tag{A.1.15}$$

Thus by induction we have proved the lemma. □

A.2 Application to the random walk kernel

Chung and Yau [27] used the diffusion kernel equivalent of lemma A.1.1 to calculate the form of the diffusion kernel on a Bethe lattice with degree d. They did this by realising that the Bethe lattice with unit weights is a covering of an infinite path graph, P, with vertex set $\mathcal{V} = \{0, 1, 2, \ldots\}$, edge set $\mathcal{E} = \{(j-1, j) | j \in \mathbb{N}\}$ and weight function $w(j-1, j) = d(d-1)^{j-1}$. Here we will discuss the equivalent steps for the random walk kernel.

Since the random walk kernel is symmetric positive definite it can be represented by the eigenvalues λ_i and orthonormal eigenvectors $\boldsymbol{\phi}_i$ of $\boldsymbol{L}$ via

$$(\boldsymbol{I} - a^{-1}\boldsymbol{L})^p = \sum_i (1 - a^{-1}\lambda_i)^p \boldsymbol{\phi}_i \boldsymbol{\phi}_i{}^{\mathrm{T}}. \tag{A.2.1}$$

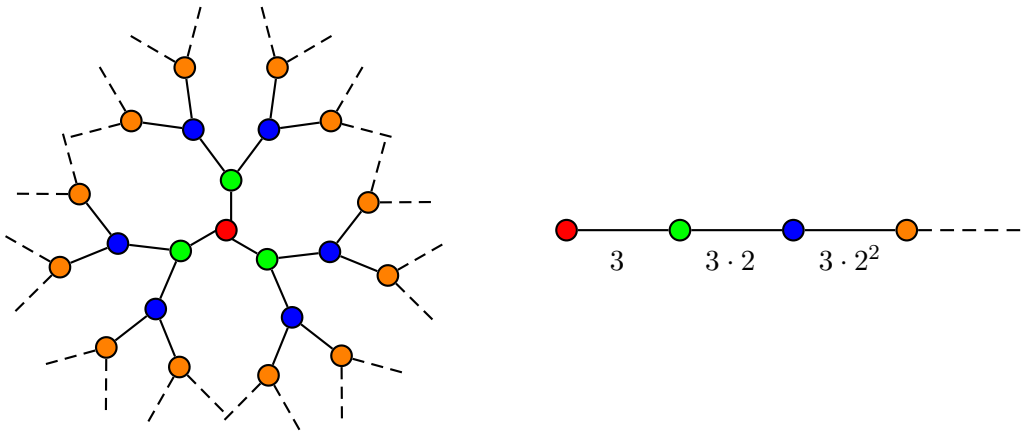


Figure A.1: The covering of a weighted path P by the Bethe lattice. Vertices in 'shells' about a vertex in the Bethe lattice are assigned to vertices in the path corresponding to their shell number. Edge weights in the path are equal to the number of vertices in the shell.

We can therefore calculate the random walk kernel if we can calculate the eigenvalues and eigenvectors of the normalised Laplacian of the graph. Chung and Yau [27] calculate the eigenvalues and eigenvectors of P by first examining the eigenvalues and eigenvectors of the truncated path up to a fixed distance, l, and then taking l to infinity. The authors found the eigenvalues of the truncated path represented by the $l \times l$ matrix

$$\begin{pmatrix}
1 & -\frac{1}{\sqrt{d}} & 0 & \cdots & \cdots & \cdots & 0 \\
-\frac{1}{\sqrt{d}} & 1 & -\frac{\sqrt{d-1}}{d} & \ddots & & & \vdots \\
0 & -\frac{\sqrt{d-1}}{d} & 1 & -\frac{\sqrt{d-1}}{d} & \ddots & & \\
\vdots & \ddots & -\frac{\sqrt{d-1}}{d} & 1 & -\frac{\sqrt{d-1}}{d} & \ddots & \vdots \\
 & & \ddots & -\frac{\sqrt{d-1}}{d} & 1 & -\frac{\sqrt{d-1}}{d} & 0 \\
 & & & \ddots & -\frac{\sqrt{d-1}}{d} & 1 & -\frac{\sqrt{d-1}}{d} \\
0 & \cdots & \cdots & \cdots & 0 & 1 & -\sqrt{\frac{d-1}{d}} \\
0 & \cdots & \cdots & \cdots & 0 & -\sqrt{\frac{d-1}{d}} & 1
\end{pmatrix} \tag{A.2.2}$$

to be $\lambda_0 = 0$, $\lambda_l = 2$ and $\lambda_j = 1 - \frac{2\sqrt{d-1}}{d} \cos \frac{\pi j}{l}$ for $j = 1, \ldots, l-1$ with corresponding

eigenvectors of the form $\boldsymbol{\phi}_j = \boldsymbol{f}_j/\|\boldsymbol{f}_j\|$ with

$$(\boldsymbol{f}_0)_p = \begin{cases} 1 & p = 0 \\ \sqrt{d(d-1)^{p-1}} & 1 \leq p \leq l-1 \\ \sqrt{(d-1)^{l-1}} & p = l \end{cases} \tag{A.2.3}$$

$$(\boldsymbol{f}_j)_p = \begin{cases} \sqrt{\dfrac{d}{d-1}} \sin\dfrac{\pi j}{l} & p = 0 \\ \sin\dfrac{\pi j(p+1)}{l} - \dfrac{1}{d-1}\sin\dfrac{\pi j(p-1)}{l} & 1 \leq p \leq l-1 \\ -\dfrac{\sqrt{d}}{d-1}\sin\dfrac{\pi j}{l} & p = l \end{cases} \tag{A.2.4}$$

$$(\boldsymbol{f}_l)_p = \begin{cases} 1 & p = 0 \\ (-1)^p\sqrt{d(d-1)^{p-1}} & 1 \leq p \leq l-1 \\ (-1)^l\sqrt{(d-1)^{l-1}} & p = l \end{cases} \tag{A.2.5}$$

Substituting the eigenvalues and eigenvectors into the random walk kernel for the truncated path and taking l to infinity we find the random walk kernel on a path to be given by

$$C_{00} = \frac{2d(d-1)}{\pi}\int_0^\pi \mathrm{d}x\, \frac{\left(1 - a^{-1}[1 - \frac{2\sqrt{d-1}}{d}\cos x]\right)^p \sin^2(x)}{d^2 - 4(d-1)\cos^2(x)} \tag{A.2.6}$$

$$C_{0l\geq 1} = \frac{2\sqrt{d(d-1)}}{\pi}\int_0^\pi \mathrm{d}x\, \left(1 - a^{-1}[1 - \frac{2\sqrt{d-1}}{d}\cos x]\right)^p \times \frac{\sin x\,[(d-1)\sin((l+1)x) - \sin((l-1)x)]}{d^2 - 4(d-1)\cos^2(x)}. \tag{A.2.7}$$

By lemma A.1.1 we have that the random walk kernel on a Bethe lattice from vertex 0 to any vertex in a shell of distance l will be given by

$$\tilde{C}_{00} = \frac{2d(d-1)}{\pi}\int_0^\pi \mathrm{d}x\, \frac{\left(1 - a^{-1}[1 - \frac{2\sqrt{d-1}}{d}\cos x]\right)^p \sin^2(x)}{d^2 - 4(d-1)\cos^2(x)} \tag{A.2.8}$$

$$\tilde{C}_{0l\geq 1} = \frac{2}{\pi(d-1)^{l/2-1}}\int_0^\pi \mathrm{d}x\, \left(1 - a^{-1}[1 - \frac{2\sqrt{d-1}}{d}\cos x]\right)^p \times \frac{\sin x\,[(d-1)\sin((l+1)x) - \sin((l-1)x)]}{d^2 - 4(d-1)\cos^2(x)}. \tag{A.2.9}$$

Since the Bethe lattice is identical seen from any starting vertex (A.2.8) and (A.2.9) define the random walk kernel completely. To calculate (3.1.3) all that is required is to

realise that $\hat{C}_{lp} = \tilde{C}_{0l}$ then $C_{lp} = \hat{C}_{l,p}/C_{0,p}$ i.e. the random walk kernel will be given by (A.2.9) divided by (A.2.8). For large p we see the main contributions to the integral will be from values that are up to $O(p^{-1/2})$. We can therefore approximate $\cos x \approx 1$ and $\sin x \approx x$ to get (3.1.3), restated below:

$$C_{l,p\to\infty} = \left[1 + \frac{l(d-2)}{d}\right] \frac{1}{(d-1)^{l/2}} \tag{A.2.10}$$

A.3 Scaling for large p

In section 4.3 we derived the learning curve decay caused by a single example, given in (4.3.1). As explained there, to understand how this behaves for large p one needs to know how $C_{l,p}$ approaches its limiting form (A.2.10). In the derivation in the previous subsection we saw that in order to obtain this limiting form one expands not just $\sin(x)$, but also $\sin((l-1)x)$ and $\sin((l+1)x)$ linearly because $x = O(p^{-1/2})$ is small. For the two sine factors this will break down once $lp^{-1/2}$ is of order one, i.e. for large enough l. This suggests looking at $C_{l,p}$ as function of $l' = lp^{-1/2}$. To focus on the relevant part of the integral one can similarly transform the integration variable to $x' = xp^{1/2}$. For p taken large at constant l', the integral for $\hat{C}_{l,p}$ in (A.2.9) then becomes proportional to

$$(d-1)^{-l/2} \int_0^\infty \mathrm{d}x'\, \mathrm{e}^{-x'^2\sqrt{d-1}/(da)} x'(d-2)\sin(l'x') \tag{A.3.1}$$

Integration by parts then gives $C_{l,p} \propto \hat{C}_{l,p} \propto (d-1)^{-l/2} l' \exp(-\frac{(l')^2 da}{4\sqrt{d-1}})$. Comparing with the large-l limit of (A.2.10), $C_{l,p} \propto (d-1)^{-l/2} l'$, shows that the effect of finite p is contained in the exponential (Gaussian) cutoff factor which becomes significant for l' of order unity, i.e. $l = O(p^{1/2})$, as stated in the main text.

Visually, the cutoff effects for large l and p can be seen more clearly by plotting not $C_{l,p}$ directly but rather $\hat{R}_{l,p}$, the unnormalised version of $R_{l,p}$ as defined in section 3.1.1. As $v_l = d(d-1)^{l-1}$ for $l \geq 1$, the additional $\sqrt{v_l}$ factor just removes the decay with $(d-1)^{-l/2}$ from $C_{l,p}$. From the above discussion, we then expect to find for large l that $\hat{R}_{l,p} \propto l \exp(-\frac{(l')^2 da}{4\sqrt{d-1}})$. A plot of numerical results for a suitably normalised version of $\hat{R}_{l,p}$ (see below), plotted against $lp^{-1/2}$ for increasing p, is shown in figure A.2. There is a clear trend towards the predicted asymptotic behaviour for large p (dashed line).

It may be somewhat surprising that the cut off lengthscale that appears in the analysis

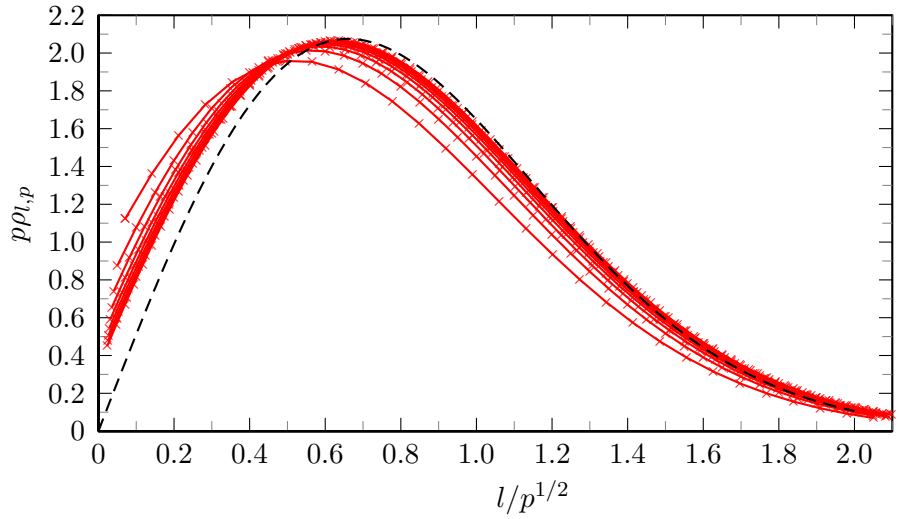


Figure A.2: Plot demonstrating the $p \to \infty$ scaling from of $p\rho_{l,p}$, the normalised version of $\hat{R}_{l,p}$ for $d = 3$ and $a = 2$. Dashed line shows the analytically found functional form of the scaling.

above is $l = O(p^{1/2})$, which for large p is much smaller than the typical kernel range p/a. To understand this intuitively, one can go back to (3.1.4) for $R_{l,p}$ and use that this is essentially unnormalised diffusion. Taking $\hat{R}_{l,p}$ as the unnormalised version of $R_{l,p}$ again and letting $\rho_{l,p} = \hat{R}_{l,p}\gamma^{-p}$ with $\gamma = (1 - 1/a) + 2\sqrt{d-1}/(ad)$ to re-establish normalisation, one finds that $\rho_{l,p}$ evolves almost according to a diffusion process in 'time' p, except that probability conservation is broken at $l = 0$ and $l = 1$. In the (leaky) diffusion process, $\rho_{l,p}$, the typical lengthscale l should then scale with time as $p^{1/2}$, exactly as we found above.

This diffusion interpretation could also be used to reproduce the quantitative scaling form, by adapting the methods of Monthus and Texier [79] where the authors study a one dimensional walk with a reflective boundary to compute the large-p scaling. The normalised version of $\hat{R}_{l,p}$ shown in figure A.2 is in fact $p\rho_{l,p}$.

APPENDIX B

REPLICA DERIVATIONS

We discuss here details of some of the derivations not shown in Chapter 5 because they would have disrupted the flow of the argument. There are two results used in the main text that are only referred to and not explicitly calculated. The first is the saddle point method for multivariate complex functions, the second is the saddle point calculation of the graph normaliser $\mathcal{N}$. Before we derive these, however, we will begin with a discussion of the saddle point approximation.

B.1 The saddle point method a.k.a. the method of steepest descents

For both the globally and locally normalised kernel replica derivations in the main text we make extensive use of the saddle point method, also known as the method of steepest descents. We will derive here the one and two dimensional version of the saddle point method and refer the reader to Kaminski [59] for further information about the multivariate case. Our derivation will begin with the saddle point method for the real line which is more commonly known as Laplace's method [13, 33, 54]. We will follow the derivation in Costin [33].

B.1.1 Laplace's Method

We begin by considering an integral of the form

$$I = \int_{-\infty}^{\infty} \mathrm{d}x\, g(x) \exp(-Vh(x)) \tag{B.1.1}$$

as V tends to infinity and where both h and g are real functions mapping from the real line to the real line. We will assume that[1]

1. g is continuous and independent of V

2. h is twice differentiable and has a unique minimum without loss of generality at $x = 0$

3. $g(0) \neq 0$ and without loss of generality $g(0) > 0$

4. $h''(0) > 0$

5. I is absolutely convergent for $V \geq 1$.

From the assumption that h and g are continuous we must have that for any $\epsilon > 0$

$$|h(x) - h(0) - \frac{1}{2}h''(0)x^2| < \epsilon x^2 \tag{B.1.2}$$

and

$$|g(x) - g(0)| < \epsilon \tag{B.1.3}$$

when $|x| < \delta(\epsilon)$ for some $\delta(\epsilon) > 0$. We also see that, since h has only one minimum, for any $\delta > 0$ there must exist a c such that $|h(x)| > c$ for all $|x - x_0| > \delta$. We use these facts to derive the Laplace result.

We begin by breaking up the integral (B.1.1) in the following way,

$$I = \int_{-\infty}^{-\delta(\epsilon)} \mathrm{d}x\, g(x) \exp(-Vh(x)) + \int_{-\delta(\epsilon)}^{\delta(\epsilon)} \mathrm{d}x\, g(x) \exp(-Vh(x)) + \int_{\delta(\epsilon)}^{\infty} \mathrm{d}x\, g(x) \exp(-Vh(x)) \tag{B.1.4}$$

[1] These conditions are stronger than is necessary to prove this result; we could for instance relax the requirement of a single minimum.

and now evaluate each of the integrals in this equation as V becomes large beginning with the two 'tail' integrals. We see that for the tail integrals the following holds

$$\begin{aligned}\int_{|\delta(\epsilon)|\geq 0} \mathrm{d}x\, g(x)\exp(-Vh(x)) &\leq \int_{|\delta(\epsilon)|\geq 0} \mathrm{d}x\, |g(x)|\exp(-(V-1)h(x)-h(x)) \\ &\leq \int_{|\delta(\epsilon)|\geq 0} \mathrm{d}x\, |g(x)|\exp(-(V-1)c-h(x)) \\ &\leq \exp(-(V-1)c)\int_{-\infty}^{\infty} \mathrm{d}x\, |g(x)|\exp(-h(x))\end{aligned}$$

This implies (by property 5.) that the tails must tend to zero exponentially at a rate at least $\exp(-(V-1)c)$. We now examine the final internal integral. Using the (B.1.2) we sandwich the integral. We see that we may bound the internal integral from above via

$$\begin{aligned}\int_{-\delta(\epsilon)}^{\delta(\epsilon)} \mathrm{d}x\, g(x)\exp(-Vh(x)) &= \int_{-\delta(\epsilon)}^{\delta(\epsilon)} \mathrm{d}x\, g(x)\exp(-V(h(x)+h(0)-h(0))) \\ &\leq (g(0)+\epsilon)\mathrm{e}^{Vh(0)}\int_{-\delta(\epsilon)}^{\delta(\epsilon)} \mathrm{d}x\, \exp\left(-\frac{V}{2}(h''(0)-\epsilon)x^2\right) \\ &\leq (g(0)+\epsilon)\mathrm{e}^{Vh(0)}\int_{-\infty}^{\infty} \mathrm{d}x\, \exp\left(-\frac{V}{2}(h''(0)-\epsilon)x^2\right) \\ &= \sqrt{\frac{2\pi}{V(h''(0)-\epsilon)}}(g(0)+\epsilon)\mathrm{e}^{Vh(0)}\end{aligned}$$

from below we may bound the integral similarly using

$$\begin{aligned}\int_{-\delta(\epsilon)}^{\delta(\epsilon)} \mathrm{d}x\, g(x)\exp(-Vh(x)) &= \int_{-\delta(\epsilon)}^{\delta(\epsilon)} \mathrm{d}x\, g(x)\exp(-V(h(x)+h(0)-h(0))) \\ &\geq (g(0)-\epsilon)\mathrm{e}^{Vh(0)}\int_{-\delta(\epsilon)}^{\delta(\epsilon)} \mathrm{d}x\, \exp\left(-\frac{V}{2}(h''(0)+\epsilon)x^2\right) \\ &\to \sqrt{\frac{2\pi}{V(h''(0)+\epsilon)}}(g(0)-\epsilon)\mathrm{e}^{Vh(0)} \qquad \text{as } V\to\infty\end{aligned}$$

Combining these two results we find we can use a sandwiching argument for the internal integral for large V. Finally if we take ϵ to zero we find that since the internal integral scales with $\mathrm{e}^{Vh(0)}/\sqrt{V}$ and the tails scale exponentially with e^{-Vc} the original integral given in (B.1.1) is dominated by the internal integral. The final Laplace method approximation is therefore given by

$$\int_{-\infty}^{\infty} \mathrm{d}x\, g(x)\exp(-Vh(x)) \approx \mathrm{e}^{Vh(0)}g(0)\sqrt{\frac{2\pi}{h''(0)V}} \tag{B.1.5}$$

for large V. The extension of this result so that the minimum occurs at a point x_0 is trivial and yields,

$$\int_{-\infty}^{\infty} \mathrm{d}x\, g(x) \exp(-Vh(x)) \approx \mathrm{e}^{Vh(x_0)} g(x_0) \sqrt{\frac{2\pi}{h''(x_0)V}} \tag{B.1.6}$$

B.1.2 Saddle-point method

We now wish to extend Laplace's method to the complex plane. We wish to approximate

$$I = \int_{\Gamma} \mathrm{d}z\, g(z) \exp(-Vh(z)) \tag{B.1.7}$$

for g and h functions that map from the complex plane to the complex plane and Γ a path in the complex plane from a point A to a point B contained within a region U where $h(z)$ and $g(z)$ are analytic. We would like to perform a similar method to Laplace's method. First, however, we need two important facts from complex analysis [32]

1. **Cauchy's theorem** which states that any closed contour integral within a space $U \subseteq \mathbb{C}$ in which the integrand is analytic is zero. This allows us to move our path Γ how we choose so long as we remain inside U (see figure B.1).

2. **Maximum modulus theorem** which states that the modulus of an analytic function has no maxima. This implies that any point in which the derivative of $h(z)$ is zero must correspond to a saddle point of $\exp(-Vh(z))$.

To apply Laplace's method we need to find a minimum z_0 of the real component of $h(z)$ along the path Γ. Cauchy's theorem tells us that instead of looking for the minimum point of of the real component of $h(z)$ only along Γ we may look for the minimum point of the real component of $h(z)$ in the space U. We may then deform our path to pass through this point without changing the resulting integral. By the maximum modulus theorem we also see that any point that minimises $\mathrm{Re}\, h(z)$, i.e. maximises the modulus of $\exp(-Vh(z))$, must be a saddle point of $\exp(-Vh(z))$ in U.

The final thing we must worry about before applying Laplace's method is the imaginary component of $\exp(-Vh(z))$ about the saddle point. If this is not constant it could lead to oscillatory cancellations in the large V limit. We must therefore choose to deform our path so that it passes through the saddle point in such a way that the imaginary

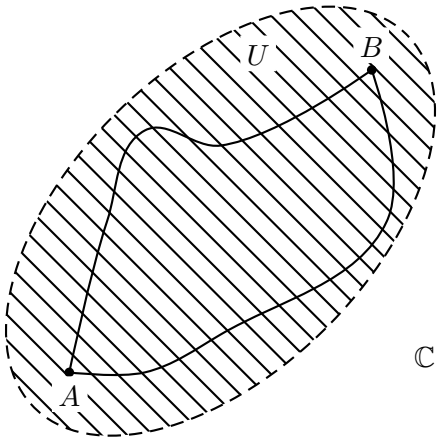


Figure B.1: An explanation of how we use Cauchy's theorem. Since the integral of any closed path is zero we see that the integrals of two different paths within U with the same end points must be equal we can therefore deform our path integral how we like so long as the path remains in U.

component is constant in a region about the saddle point. If we pick our path in this way then we see that the Taylor expansion of h along the path about the saddle point z_0 will be

$$h(z) \approx h(z_0) + \frac{1}{2}h''(z_0)(z - z_0)^2 \tag{B.1.8}$$

with $\frac{1}{2}h''(z_0)(z-z_0)^2$ real and positive. This will mean that the arguments of the second order term must have the property that

$$\arg(\frac{1}{2}h''(z_0)(z - z_0)^2) = \arg(h''(z_0)) + 2\arg(z - z_0) = 0 \tag{B.1.9}$$

which implies our path must cross z_0 in the direction $\arg(z - z_0) = -\frac{1}{2}\arg(h''(z_0)) = \theta$. We can now apply Laplace's method to our integral (B.1.7). We deform our path so that we may break the integral I into

$$\int_\Gamma \mathrm{d}z\, g(z)\exp(-Vh(z)) = \int_{\Gamma_A} \mathrm{d}z\, g(z)\exp(-Vh(z)) + \int_{\Gamma_{SP}} \mathrm{d}z\, g(z)\exp(-Vh(z)) + \int_{\Gamma_B} \mathrm{d}z\, g(z)\exp(-Vh(z)) \tag{B.1.10}$$

where Γ_A and Γ_B are paths which get us from the start and end points to the saddle point region and Γ_{SP} is the path in the saddle point region that satisfies the previously mentioned properties (see figure B.2).

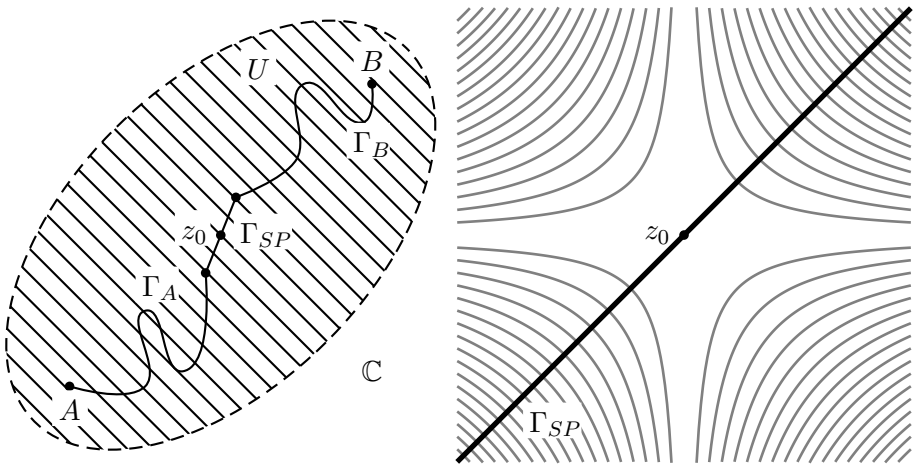


Figure B.2: A diagram of the deformed path used for calculating the asymptotics of equation (B.1.7)

With the integral broken down in this way we see that the two 'tails' Γ_A and Γ_B exponentially decay and we are left with the integral over Γ_{SP} which we may approximate using Laplace's method. This results in the saddle point approximation given by

$$\begin{aligned}\int_\Gamma \mathrm{d}z\, g(z)\exp(-Vh(z)) &\approx \mathrm{e}^{-\mathrm{i}\theta}\mathrm{e}^{-Vh(z_0)}g(z_0)\sqrt{\frac{2\pi}{h''(z_0)V}} \\ &= \mathrm{e}^{-Vh(z_0)}g(z_0)\sqrt{\frac{2\pi}{|h''(z_0)|V}} \end{aligned} \tag{B.1.11}$$

for large V.

The multivariate version of (B.1.11) is complicated to derive and suffers from the problem that the path of steepest descent is possibly not unique [59]. As a consequence of this there are a few ways of picking one's path (manifold) for applying the saddle point approximation. Before discussing one way of deforming a manifold, however, we require a $\mathbb{C}^n$ equivalent of Cauchy's theorem for deforming path integrals in $\mathbb{C}$. This is given by Fedoryuk [41] in Russian and later translated into English in Kaminski [59].

Theorem B.1.1 (Manifold deformation [41, 59])**.** Let γ be a compact smooth n-manifold (possibly with a boundary) and let h and g be functions holomorphic at $\boldsymbol{z}^0 \in \gamma$. Among the points at which $\min\{\mathrm{Re}\, h(\boldsymbol{z}) | \boldsymbol{z} \in \gamma\}$ is attained, suppose there are no saddle points of h and no boundary points of γ. Then there exists a manifold γ_1 such that

1. $\int_\gamma \mathrm{d}\boldsymbol{z} g(\boldsymbol{z})\exp(-\lambda h(\boldsymbol{z})) = \int_{\gamma_1} \mathrm{d}\boldsymbol{z} g(\boldsymbol{z})\exp(-\lambda h(\boldsymbol{z}))$

2. $\min_{\boldsymbol{z}\in\gamma_1} \operatorname{Re} h(\boldsymbol{z}) < \min_{\boldsymbol{z}\in\gamma} \operatorname{Re} h(\boldsymbol{z})$

With this theorem we now see that we may deform a manifold and apply a n-dimensional saddle point approximation. We shall now discuss one way of deforming the manifold as given in Kaminski [59] so that we may use a saddle point approximation. This method is called the *coordinate-wise descent surface construction*. As in Kaminski [59] we will focus on the 2-dimensional case, which is all we require in the main text.

We will assume that $\boldsymbol{z}_0 = (u_0, v_0)$ is the saddle point of $h(u, v)$ for $u, v \in \mathbb{C}$, and consider the function $\tilde{h}(s,t) = h(u_0 + s\mathrm{e}^{\mathrm{i}\theta}, v_0 + t\mathrm{e}^{\mathrm{i}\phi}) - h(\boldsymbol{z}_0)$ where θ and ϕ are picked so that $h(u_0 + s\mathrm{e}^{\mathrm{i}\theta}, v_0 + t\mathrm{e}^{\mathrm{i}\phi})$ has constant imaginary component about $\boldsymbol{z}_0$ resulting in $\tilde{h}(s,t)$ being real in a region around $(0,0)$. We see then that near $(0,0)$

$$\tilde{h}(s,t) \approx \frac{1}{2}(s,t)\tilde{\boldsymbol{H}}(0,0)(s,t)^{\mathrm{T}} \tag{B.1.12}$$

where $\tilde{\boldsymbol{H}}(0,0)$ is the Hessian of $\tilde{h}$ evaluated at the saddle point. Similarly to the one dimensional saddle point method we want to apply the Laplace approximation at the saddle point. We therefore require that $\tilde{\boldsymbol{H}}(0,0)$ is positive (semi-)definite. Using Sylvester's criterion we see that this will occur if and only if the determinant of all the principal submatrices of $\tilde{\boldsymbol{H}}(0,0)$ are positive [21]. For the two dimensional case this corresponds to requiring that $(\tilde{\boldsymbol{H}}(0,0))_{11} > 0$ and $\det \tilde{\boldsymbol{H}}(0,0) > 0$. These two conditions give us a set of equations which we can solve to calculate θ and ϕ, the direction of the surface through the saddle point. We see that

$$(\tilde{\boldsymbol{H}}(0,0))_{11} = \mathrm{e}^{2\mathrm{i}\theta}(\boldsymbol{H}(\boldsymbol{z}_0))_{11} \tag{B.1.13}$$

$$\det \tilde{\boldsymbol{H}}(0,0) = \mathrm{e}^{2\mathrm{i}\theta}\mathrm{e}^{2\mathrm{i}\phi} \det \boldsymbol{H}(\boldsymbol{z}_0) \tag{B.1.14}$$

which implies

$$\theta = -\frac{1}{2}\arg(\boldsymbol{H}(\boldsymbol{z}_0))_{11} \tag{B.1.15}$$

$$\phi = \frac{1}{2}\arg(\boldsymbol{H}(\boldsymbol{z}_0))_{11} - \frac{1}{2}\arg\det \boldsymbol{H}(\boldsymbol{z}_0) \tag{B.1.16}$$

Hence for functions that map from $\mathbb{C}^2$ to the complex plane a saddle point approximation is given by

$$\begin{aligned}\int \mathrm{d}\boldsymbol{z} g(\boldsymbol{z}) \exp(-Vh(\boldsymbol{z})) &\approx \left(\frac{2\pi}{V}\right)^{n/2} g(\boldsymbol{z}_0)\mathrm{e}^{-\mathrm{i}(\theta+\phi)}\mathrm{e}^{-Vh(\boldsymbol{z}_0)} \left[\det \tilde{\boldsymbol{H}}(\boldsymbol{z}_0)\right]^{-1/2} \\ &= \left(\frac{2\pi}{V}\right)^{n/2} g(\boldsymbol{z}_0)\mathrm{e}^{-Vh(\boldsymbol{z}_0)} \left|\det \boldsymbol{H}(\boldsymbol{z}_0)\right|^{-1/2}\end{aligned} \tag{B.1.17}$$

for $(\boldsymbol{H}(\boldsymbol{z}))_{ij} = \frac{\partial^2 h}{\partial z_i \partial z_j}$, the second order differentials in the Taylor expansion of $h(\boldsymbol{z})$ about the saddle point $\boldsymbol{z}_0$. We use this result in the main text around (5.2.23).

The final saddle point based result we use in the main text is to calculate the generalisation error, ϵ_g, given in (5.1.39) and (5.2.35) for large V. To calculate the generalisation error we need to calculate the large V limit of a function of the form

$$-\frac{2}{V}\frac{\partial}{\partial\lambda}\log\int \mathrm{d}\boldsymbol{z}\exp\left(-V(f(\boldsymbol{z})+g(\boldsymbol{z},\lambda))\right) \tag{B.1.18}$$

which after differentiation yields,

$$\frac{\int \mathrm{d}\boldsymbol{z}\, 2\left[\frac{\partial}{\partial\lambda}g(\boldsymbol{z},\lambda)\right]\exp\left(-V(f(\boldsymbol{z})+g(\boldsymbol{z},\lambda))\right)}{\int \mathrm{d}\boldsymbol{z}\,\exp\left(-V(f(\boldsymbol{z})+g(\boldsymbol{z},\lambda))\right)} \tag{B.1.19}$$

If we apply (B.1.17) to the numerator and denominator then we see this fraction simplifies to be just $2\frac{\partial g(\boldsymbol{z}_0,\lambda)}{\partial\lambda}$ for $\boldsymbol{z}_0$ the saddle point of $f(\boldsymbol{z})+g(\boldsymbol{z},\lambda)$

B.2 Graph Normaliser

In this section we derive the large V asymptotics of the graph ensemble normaliser, $\mathcal{N}$, in (5.1.5) and (5.2.8) and show that it cancels with the $O(n^0)$ terms of the exponent of (5.1.20).

The graph normaliser is defined as a sum over all possible graphs G; i.e. over all adjacency matrices with elements $a_{ij} = a_{ji}$

$$\mathcal{N} = \sum_G \prod_{i<j}\left[\frac{\bar{d}}{V}\delta_{a_{ij},1} + \left(1-\frac{\bar{d}}{V}\right)\delta_{a_{ij},0}\right]\prod_i \delta_{d_i,\sum_j a_{ij}} \tag{B.2.1}$$

$$= \int_0^{2\pi}\prod_i \frac{\mathrm{d}\hat{d}_i}{2\pi}\mathrm{e}^{-\mathrm{i}\hat{d}_i d_i}\prod_{i<j}\sum_{a_{ij}\in\{0,1\}}\left[\frac{\bar{d}}{V}\delta_{a_{ij},1} + \left(1-\frac{\bar{d}}{V}\right)\delta_{a_{ij},0}\right]\mathrm{e}^{\mathrm{i}a_{ij}(\hat{d}_i+\hat{d}_j)} \tag{B.2.2}$$

$$= \int_0^{2\pi}\prod_i \frac{\mathrm{d}\hat{d}_i}{2\pi}\mathrm{e}^{-\mathrm{i}\hat{d}_i d_i}\prod_{i,j}\exp\left(\frac{\bar{d}}{2V}\left(\mathrm{e}^{\mathrm{i}(\hat{d}_i+\hat{d}_j)}-1\right)\right), \tag{B.2.3}$$

where going from (B.2.2) to (B.2.3) we have assumed that V is large so that $\exp(\frac{x}{V}) \approx 1+\frac{x}{V}$. Defining the density $\rho_0 = \frac{1}{V}\sum_i \mathrm{e}^{\mathrm{i}\hat{d}_i}$ with a conjugate enforcement term $\hat{\rho}_0$, (B.2.3) becomes,

$$\mathcal{N} = \int \mathrm{d}\rho_0 \mathrm{d}\hat{\rho}_0 \exp\left(V\left[\frac{\bar{d}}{2}(\rho_0^2-1) - \mathrm{i}\rho_0\hat{\rho}_0\right]\right)\prod_i\int_0^{2\pi}\frac{\mathrm{d}\hat{d}_i}{2\pi}\mathrm{e}^{-\mathrm{i}d_i\hat{d}_i}\mathrm{e}^{\mathrm{i}\hat{\rho}_0\sum_i\exp(\mathrm{i}\hat{d}_i)}. \tag{B.2.4}$$

Rewriting the final exponential in terms of its power series and integrating over $\hat{d}_i$ gives

$$\mathcal{N} = \int \mathrm{d}\rho_0 \mathrm{d}\hat{\rho}_0 \exp\left(V\left[\frac{\bar{d}}{2}\{\rho_0^2 - 1\} - \mathrm{i}\rho_0\hat{\rho}_0\right]\right)\prod_i \frac{(\mathrm{i}\hat{\rho}_0)^{d_i}}{d_i!}. \tag{B.2.5}$$

Defining the degree distribution $p(d) = (1/V)\sum_i \delta_{d_i,d}$ in a similar manner to (5.1.18) and bringing the final term into the exponent of the first gives us a form amenable to saddle point evaluation:

$$\mathcal{N} = \int \mathrm{d}\rho_0 \mathrm{d}\hat{\rho}_0 \exp\left(V\left[\frac{\bar{d}}{2}(\rho_0^2 - 1) - \mathrm{i}\rho_0\hat{\rho}_0 + \sum_d p(d)\log\frac{(\mathrm{i}\hat{\rho}_0)^d}{d!}\right]\right). \tag{B.2.6}$$

The saddle point of the exponent occurs when

$$0 = \bar{d}\rho_0 - \mathrm{i}\hat{\rho}_0 \tag{B.2.7}$$

$$0 = \sum_d p(d) d\hat{\rho}_0^{-1} - \mathrm{i}\rho_0 \tag{B.2.8}$$

and solving gives $\rho_0 = 1$ and $\hat{\rho}_0 = -\mathrm{i}\bar{d}$.

We now consider the $O(1)$ terms of the exponent of (5.1.20). We see that to leading order we have

$$\int \mathcal{D}\rho\mathcal{D}\hat{\rho}\, \exp\left(V\left[\left(\frac{\bar{d}}{2}\int \mathcal{D}\psi\mathcal{D}\psi'\pi[\psi]\pi[\psi'] - 1\right) - \bar{d}\int \mathcal{D}\psi\mathcal{D}\hat{\psi}\pi[\psi]\hat{\pi}[\hat{\psi}] \right.\right.$$
$$\left.\left. + \sum_d p(d)\log\left(\frac{\bar{d}^d}{d!}\int\prod_{i=1}^d \mathcal{D}\hat{\psi}^i\hat{\pi}[\hat{\psi}^i]\right)\right]\right) \tag{B.2.9}$$

which we will denote by $\mathcal{T}$. We require the saddle points of $\mathcal{T}$ with respect to ρ and $\hat{\rho}$; these occur when[2]

$$0 = \bar{d}\int \mathcal{D}\psi'\pi[\psi'] - \bar{d}\int \mathcal{D}\hat{\psi}\hat{\pi}[\hat{\psi}] \tag{B.2.10}$$

$$0 = \bar{d}\int \mathcal{D}\psi\pi[\psi] + \sum_d dp(d)\frac{\int\prod_{i=1}^{d-1}\mathcal{D}\hat{\psi}^i\hat{\pi}[\hat{\psi}^i]}{\int\prod_{i=1}^{d}\mathcal{D}\hat{\psi}^i\hat{\pi}[\hat{\psi}^i]} \tag{B.2.11}$$

which when rearranged gives us

$$\int \mathcal{D}\psi\pi[\psi] = \int \mathcal{D}\hat{\psi}\hat{\pi}[\hat{\psi}] \tag{B.2.12}$$

$$\int \mathcal{D}\psi\pi[\psi] = \sum_d \frac{dp(d)}{\bar{d}}\frac{\int\prod_{i=1}^{d-1}\mathcal{D}\psi^i\pi[\psi^i]}{\int\mathcal{D}\psi^i\pi[\psi^i]} = \frac{1}{\int\mathcal{D}\psi\pi[\psi]}\sum_d\frac{dp(d)}{\bar{d}} = \frac{1}{\int\mathcal{D}\psi\pi[\psi]} \tag{B.2.13}$$

[2]Note that unlike in the main text we do not require Lagrange multipliers since at this stage there is no constraint on the form of $\pi[\psi]$ and $\hat{\pi}[\hat{\psi}]$.

Equations (B.2.10) and (B.2.11) are therefore solved by $\int \mathcal{D}\psi\pi[\psi] = 1$ and $\int \mathcal{D}\hat{\psi}\hat{\pi}[\hat{\psi}] = 1$ which explains the choice of prefactor in (5.2.13). Finally to see that the saddle point approximations of $\mathcal{N}$ and $\mathcal{T}$ cancel we substitute the saddle point values into (B.2.6) and (B.2.9). Since, apart from a rescaling of $\hat{\rho}_0$ by $-\mathrm{i}\bar{d}$, they have the same functional form in the exponent we see that $\mathcal{T}/\mathcal{N}$ will equal one in the limit of large V.

B.3 Large L saddle point approximation for ϵ_g

In this section we discuss the derivation of (5.2.35) in the main text. We begin with (5.2.20)

$$\epsilon_g = \lim_{\lambda\to 0}\sum_d p(d)\left\langle \int\prod_{i=1}^{d}\mathcal{D}\hat{\psi}^i\hat{\pi}[\hat{\psi}^i]\int \mathrm{d}\boldsymbol{r}\frac{\mathrm{d}\kappa\mathrm{d}\hat{\kappa}L}{2\pi}\prod_{l=1}^{L}\mathrm{d}\boldsymbol{r}^l_{\mathrm{aux}} \left(\frac{1}{\gamma/\sigma^2+\lambda}-\frac{\kappa d{\boldsymbol{e}_0}^{\mathrm{T}}\boldsymbol{r}\boldsymbol{r}^{\mathrm{T}}\boldsymbol{e}_0}{(\gamma/\sigma^2+\lambda)^2}\right) q(\boldsymbol{r},\kappa,\hat{\kappa},\{\boldsymbol{r}^l_{\mathrm{aux}}\})\right\rangle_{\gamma}, \quad \text{(B.3.1)}$$

where

$$q(\boldsymbol{r},\kappa,\hat{\kappa},\{\boldsymbol{r}^l_{\mathrm{aux}}\}) \propto \exp\left(-\Psi[\hat{\Psi}[\psi^1],\ldots,\hat{\Psi}[\psi^d]](\boldsymbol{r},\kappa,\hat{\kappa},\{\boldsymbol{r}^l_{\mathrm{aux}}\})\right) \quad \text{(B.3.2)}$$

Taking advantage of the fact that $\hat{\Psi}$ is independent of κ and $\hat{\kappa}$ and utilising the notation of Chapter 5 we write

$$q(\boldsymbol{r},\kappa,\hat{\kappa},\{\boldsymbol{r}^l_{\mathrm{aux}}\}) \propto \exp\left(-\sum_{i=1}^{d}\hat{\phi}(\boldsymbol{r},\{\boldsymbol{r}^l_{\mathrm{aux}}\}) + \mathrm{i}L\kappa\hat{\kappa} - \mathcal{H}(\boldsymbol{r},d,\kappa,\gamma) - \sum_l \mathcal{H}_{\mathrm{aux}}(\boldsymbol{r}^l_{\mathrm{aux}},d,\hat{\kappa})\right) \quad \text{(B.3.3)}$$

Similar to the large L saddle approximation for the update equations (see around (5.2.23)) we wish to make a saddle point approximation for large L with respect to κ and $\hat{\kappa}$. We see that the relevant terms for the saddle point approximation are

$$\int \frac{\mathrm{d}\kappa\mathrm{d}\hat{\kappa}L}{2\pi}\left(\frac{1}{\gamma/\sigma^2+\lambda}-\frac{\kappa d{\boldsymbol{e}_0}^{\mathrm{T}}\boldsymbol{r}\boldsymbol{r}^{\mathrm{T}}\boldsymbol{e}_0}{(\gamma/\sigma^2+\lambda)^2}\right) q(\boldsymbol{r},\kappa,\hat{\kappa},\{\boldsymbol{r}^l_{\mathrm{aux}}\}) \quad \text{(B.3.4)}$$

which can be written in the form

$$\int \frac{\mathrm{d}\kappa\mathrm{d}\hat{\kappa}L}{2\pi}\left(\frac{1}{\gamma/\sigma^2+\lambda}-\frac{\kappa d{\boldsymbol{e}_0}^{\mathrm{T}}\boldsymbol{r}\boldsymbol{r}^{\mathrm{T}}\boldsymbol{e}_0}{(\gamma/\sigma^2+\lambda)^2}\right)\zeta(\boldsymbol{r},\kappa,\hat{\kappa},\{\boldsymbol{r}^l_{\mathrm{aux}}\}) \times \exp\left(-L\xi(\{\boldsymbol{r}^l_{\mathrm{aux}}\},\kappa,\hat{\kappa})\right) \quad \text{(B.3.5)}$$

where ζ and ξ are defined in (5.2.24) and (5.2.25) and where we have extended the sum over $\hat{\phi}^i$ to d in the definition of ζ. Using the methods of section B.1.2 and the Hessian given in (5.2.30) we see that the saddle point approximation of (B.3.4) is given by

$$\left(\frac{1}{\gamma/\sigma^2+\lambda} - \frac{v(\boldsymbol{H})d\boldsymbol{e}_0{}^{\mathrm{T}}\boldsymbol{r}\boldsymbol{r}^{\mathrm{T}}\boldsymbol{e}_0}{(\gamma/\sigma^2+\lambda)^2}\right)\zeta(\boldsymbol{r}, v(\boldsymbol{H}), 0, \{\boldsymbol{r}^l_{\mathrm{aux}}\}) \times \exp\left(-L\xi(\{\boldsymbol{r}^l_{\mathrm{aux}}\}, v(\boldsymbol{H}), 0)\right) \quad \text{(B.3.6)}$$

Substituting this back into (B.3.1) and collecting terms then gives (5.2.35).

APPENDIX C

MISMATCH CAVITY DERIVATIONS

We discuss in this appendix in more detail some of the steps in the main text relating to the cavity derivation of the generalisation error approximation with model mismatch.

C.1 Integration of (6.3.7)

In this section we derive in more detail the integral over $\boldsymbol{h}$ in (6.3.6). The relevant $\boldsymbol{h}$ terms in this equation are given by

$$\int \mathrm{d}\boldsymbol{h} \exp\left(-\frac{\sigma_f^2}{2}\boldsymbol{h}^{\mathrm{T}}\boldsymbol{h} + \frac{\sigma_f^2\sigma_g^2}{2(\sigma_f^2+\sigma_g^2)}\boldsymbol{h}^{\mathrm{T}}\boldsymbol{h}\right) \prod_i \delta\left(h_{\boldsymbol{X}_i} - \sum_\mu \delta_{x_\mu,i} h_\mu\right) \tag{C.1.1}$$

After simplification, and the insertion of a constant prefactor $C = \left(\frac{\sigma_f^2+\sigma_g^2}{(2\pi)\sigma_f^4}\right)^{1/2}$ that will cancel after differentiation of the partition function with respect to λ, this becomes

$$C\int \mathrm{d}\boldsymbol{h} \exp\left(-\frac{\sigma_f^4}{2(\sigma_f^2+\sigma_g^2)}\boldsymbol{h}^{\mathrm{T}}\boldsymbol{h}\right) \prod_i \delta\left(h_{\boldsymbol{X}_i} - \sum_\mu \delta_{x_\mu,i} h_\mu\right) \tag{C.1.2}$$

This is just the integration of a multidimensional Gaussian along a hyperplane in $\mathbb{R}^N$ which will itself be another Gaussian in terms of $h_{\boldsymbol{X}_i}$ for $i = 1, \ldots, V$ (see figure C.1). In order to calculate this integral we therefore only need to calculate the mean and

covariance. We see that the mean is given by

$$\begin{aligned} C\int \mathrm{d}h_{\boldsymbol{X}_i}\, h_{\boldsymbol{X}_i} \int \mathrm{d}\boldsymbol{h} \exp\left(-\frac{\sigma_f^4}{2(\sigma_f^2+\sigma_g^2)}\boldsymbol{h}^{\mathrm{T}}\boldsymbol{h}\right)\prod_k \delta\left(h_{\boldsymbol{X}_k} - \sum_\mu \delta_{x_\mu,k} h_\mu\right) \\ = \sum_\mu \delta_{x_\mu,i} C \int \mathrm{d}\boldsymbol{h}\, h_\mu \exp\left(-\frac{\sigma_f^4}{2(\sigma_f^2+\sigma_g^2)}\boldsymbol{h}^{\mathrm{T}}\boldsymbol{h}\right) \\ = 0 \end{aligned} \tag{C.1.3}$$

and the ij-th element of the covariance is given by

$$\begin{aligned} \int \mathrm{d}h_{\boldsymbol{X}_i}\mathrm{d}h_{\boldsymbol{X}_j}\, h_{\boldsymbol{X}_i} h_{\boldsymbol{X}_j} C \int \mathrm{d}\boldsymbol{h} \exp\left(-\frac{\sigma_f^4}{2(\sigma_f^2+\sigma_g^2)}\boldsymbol{h}^{\mathrm{T}}\boldsymbol{h}\right)\prod_k \delta\left(h_{\boldsymbol{X}_k} - \sum_\mu \delta_{x_\mu,k} h_\mu\right) \\ = \sum_\mu \delta_{x_\mu,j} \sum_\nu \delta_{x_\nu,i} C \int \mathrm{d}\boldsymbol{h}\, h_\mu h_\nu \exp\left(-\frac{\sigma_f^4}{2(\sigma_f^2+\sigma_g^2)}\boldsymbol{h}^{\mathrm{T}}\boldsymbol{h}\right) \\ = \frac{\sigma_f^2+\sigma_g^2}{\sigma_f^4}\sum_\mu \delta_{x_\mu,j}\sum_\nu \delta_{x_\nu,i}\delta_{\nu,\mu} = \frac{\gamma_i(\sigma_f^2+\sigma_g^2)}{\sigma_f^4}\delta_{ij} \end{aligned} \tag{C.1.4}$$

Thus the integration will give

$$\begin{aligned} C\int \mathrm{d}\boldsymbol{h} \exp\left(-\frac{\sigma_f^4}{2(\sigma_f^2+\sigma_g^2)}\boldsymbol{h}^{\mathrm{T}}\boldsymbol{h}\right)\prod_i \delta\left(h_{\boldsymbol{X}_i} - \sum_\mu \delta_{x_\mu,i} h_\mu\right) \\ = \exp\left(-\frac{\sigma_f^4}{2(\sigma_f^2+\sigma_g^2)}\sum_i \frac{1}{\gamma_i}(h_{\boldsymbol{X}_i})^2\right) \end{aligned} \tag{C.1.5}$$

In the main text we ignore the constant C because, as explained above, this will cancel after differentiation of the logarithm of the partition function, Z, with respect to λ.

C.2 Deriving the final graphical model form

In this section we perform the final steps needed to derive (6.3.10) from (6.3.9). These steps essentially mirror the steps taken in the matched scenario between (4.2.4) and

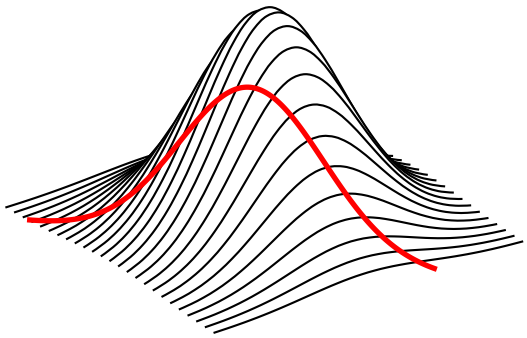


Figure C.1: A diagram to explain the integration over $\boldsymbol{h}$ resulting in (C.1.5). Any linear constraint on a Gaussian is itself another Gaussian (red line).

(4.2.6). We start with (6.3.9), which was given by

$$\begin{aligned} Z = \int \mathrm{d}\boldsymbol{f}\mathrm{d}\boldsymbol{\phi}\mathrm{d}\boldsymbol{g}\mathrm{d}\boldsymbol{\xi} \prod_i \mathrm{d}h_{\boldsymbol{X}_i} \exp\Bigg(&- \frac{1}{2}\boldsymbol{\phi}^{\mathrm{T}}\boldsymbol{C}_f\boldsymbol{\phi} - \frac{1}{2}\sum_{i=1}^{V}\left(\frac{\gamma_i}{\sigma_f^2+\sigma_g^2}+\lambda\right) f_i^2 \\ &- \frac{1}{2}\sum_{i=1}^{V}\left(\frac{\gamma_i}{\sigma_f^2+\sigma_g^2}+\lambda\right) g_i^2 - \frac{1}{2}\boldsymbol{\xi}^{\mathrm{T}}\boldsymbol{C}_g\boldsymbol{\xi} - \frac{1}{2}\sum_{i,j} h_{\boldsymbol{X}_i}(\boldsymbol{C}_f)_{ij}h_{\boldsymbol{X}_j} \\ &+ \frac{\sigma_g^2}{\sigma_f^2+\sigma_g^2}\sum_{i=1}^{V} f_i h_{\boldsymbol{X}_i} + \frac{\sigma_f^2}{\sigma_f^2+\sigma_g^2}\sum_{i=1}^{V} g_i h_{\boldsymbol{X}_i} + \sum_{i=1}^{V}\left(\frac{\gamma_i}{\sigma_f^2+\sigma_g^2}+\lambda\right) f_i g_i \\ &+ \mathrm{i}\boldsymbol{\phi}^{\mathrm{T}}\boldsymbol{f} + \mathrm{i}\boldsymbol{\xi}^{\mathrm{T}}\boldsymbol{g} - \frac{\sigma_f^4}{2(\sigma_f^2+\sigma_g^2)}\sum_i \frac{1}{\gamma_i}(h_{\boldsymbol{X}_i})^2\Bigg). \end{aligned} \tag{C.2.1}$$

Similarly to section 4.2.1 we define

$$\boldsymbol{\phi}^q = \boldsymbol{\mathcal{K}}_f^{1/2}\boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2}\boldsymbol{\mathcal{K}}_f^{-1/2}\boldsymbol{\phi}^{q-1} \tag{C.2.2}$$

$$\boldsymbol{\xi}^q = \boldsymbol{\mathcal{K}}_g^{1/2}\boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2}\boldsymbol{\mathcal{K}}_g^{-1/2}\boldsymbol{\xi}^{q-1} \tag{C.2.3}$$

$$(h^q_{\boldsymbol{X}_1},\ldots,h^q_{\boldsymbol{X}_V})^{\mathrm{T}} = \boldsymbol{\mathcal{K}}_f^{1/2}\boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2}\boldsymbol{\mathcal{K}}_f^{-1/2}(h^{q-1}_{\boldsymbol{X}_1},\ldots,h^{q-1}_{\boldsymbol{X}_V})^{\mathrm{T}} \tag{C.2.4}$$

and for convenience relabel $\boldsymbol{\phi}$ to $\boldsymbol{\phi}^0$, $\boldsymbol{\xi}$ to $\boldsymbol{\xi}^0$ and $h_{\boldsymbol{X}_i}$ to $h^0_{\boldsymbol{X}_i}$. Enforcing (C.2.2) to (C.2.4) using delta functions written in their Fourier form we see that we have

$$
\begin{aligned}
Z = \int \mathrm{d}\boldsymbol{f}\mathrm{d}\boldsymbol{g}\mathrm{d}\boldsymbol{\phi}^0 \prod_{q=1}^{p_f}\Big(\mathrm{d}\boldsymbol{\phi}^q\mathrm{d}\hat{\boldsymbol{\phi}}^q\Big)\,\mathrm{d}\boldsymbol{\xi}^0 \prod_{q=1}^{p_g}\Big(\mathrm{d}\boldsymbol{\xi}^q\mathrm{d}\hat{\boldsymbol{\xi}}^q\Big)\prod_i\left(\mathrm{d}h^0_{\boldsymbol{X}_i}\prod_{q=1}^{p_f}\Big(\mathrm{d}h^q_{\boldsymbol{X}_i}\mathrm{d}h^q_{\boldsymbol{X}_i}\Big)\right)\\
\exp\left(-\frac{1}{2}\sum_{q=0}^{p_f}c_{f,q}\boldsymbol{\phi}^{0\mathrm{T}}\boldsymbol{\mathcal{K}}_f^{-1}\boldsymbol{\phi}^q-\frac{1}{2}\sum_{i=1}^{V}\left(\frac{\gamma_i}{\sigma_f^2+\sigma_g^2}+\lambda\right)f_i^2\right.\\
-\frac{1}{2}\sum_{i=1}^{V}\left(\frac{\gamma_i}{\sigma_f^2+\sigma_g^2}+\lambda\right)g_i^2-\frac{1}{2}\sum_{q=1}^{p_g}c_{g,q}\boldsymbol{\xi}^{0\mathrm{T}}\boldsymbol{\mathcal{K}}_g\boldsymbol{\xi}^q-\frac{1}{2}\sum_i c_{f,q}\kappa_{i,f}^{-1}h^0_{\boldsymbol{X}_i}h^q_{\boldsymbol{X}_i}\\
+\frac{\sigma_g^2}{\sigma_f^2+\sigma_g^2}\sum_{i=1}^{V}f_ih^0_{\boldsymbol{X}_i}+\frac{\sigma_f^2}{\sigma_f^2+\sigma_g^2}\sum_{i=1}^{V}g_ih^0_{\boldsymbol{X}_i}+\sum_{i=1}^{V}\left(\frac{\gamma_i}{\sigma_f^2+\sigma_g^2}+\lambda\right)f_ig_i\\
+\mathrm{i}\boldsymbol{\phi}^{0\mathrm{T}}\boldsymbol{f}+\mathrm{i}\boldsymbol{\xi}^{0\mathrm{T}}\boldsymbol{g}-\frac{\sigma_f^4}{2(\sigma_f^2+\sigma_g^2)}\sum_i\frac{1}{\gamma_i}(h_{\boldsymbol{X}_i})^2+\mathrm{i}\sum_{q=1}^{p_f}(\boldsymbol{\phi}^q)^{\mathrm{T}}\hat{\boldsymbol{\phi}}^q\\
\left.+\mathrm{i}\sum_{q=1}^{p_g}(\boldsymbol{\xi}^q)^{\mathrm{T}}\hat{\boldsymbol{\xi}}^q+\mathrm{i}\sum_{i=1}^{V}\sum_{q=1}^{p_f}h^q_{\boldsymbol{X}_i}\hat{h}^q_{\boldsymbol{X}_i}\right)\\
\times\exp\left(-\mathrm{i}\sum_{q=1}^{p_f}(\hat{\boldsymbol{\phi}}^q)^{\mathrm{T}}\boldsymbol{\mathcal{K}}_f^{1/2}\boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2}\boldsymbol{\mathcal{K}}_f^{-1/2}\boldsymbol{\phi}^{q-1}\right.\\
\left.-\mathrm{i}\sum_{q=1}^{p_g}(\hat{\boldsymbol{\xi}}^q)^{\mathrm{T}}\boldsymbol{\mathcal{K}}_g^{1/2}\boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2}\boldsymbol{\mathcal{K}}_g^{-1/2}\boldsymbol{\xi}^{q-1}-\mathrm{i}\sum_{i,j}\sum_{q=1}^{p_f}\frac{\hat{h}^q_{\boldsymbol{X}_i}a_{ij}\kappa_{f,i}^{1/2}h^{q-1}_{\boldsymbol{X}_i}}{d_i^{1/2}d_j^{1/2}\kappa_{f,j}^{1/2}}\right).
\end{aligned}
\tag{C.2.5}
$$

where we have taken advantage of the binomial form of the random walk kernel given in (3.1.1) to rewrite the covariance terms in the exponent and, where $c_{f,q}$ and $c_{g,q}$ are defined, as in the main text, as the binomial prefactors $c_{f,q}=\binom{p_f}{q}(1-a_f^{-1})^{p_f-q}(a_f^q)^{-1}$ and $c_{g,q}=\binom{p_g}{q}(1-a_g^{-1})^{p_g-q}(a_g^q)^{-1}$.

Using the fact that the kernel is normalised globally so that $\kappa_{i,f}=\kappa_f$ and $\kappa_{i,g}=\kappa_g$ are constant for all i and rescaling ϕ_i^q to $\phi^q d_i^{1/2}\kappa_f^{1/2}$, $\hat{\phi}_i^q$ to $\hat{\phi}^q d_i^{1/2}\kappa_f^{-1/2}$, ξ^q to $\xi^q d_i^{1/2}\kappa_g^{1/2}$, $\hat{\xi}_i^q$ to $\hat{\xi}^q d_i^{1/2}\kappa_g^{-1/2}$, $h_{\boldsymbol{X}_i}$ to $h^q_{\boldsymbol{X}_i}d_i^{1/2}\kappa_f^{1/2}$ and $\hat{h}^q_{\boldsymbol{X}_i}$ to $\hat{h}^q_{\boldsymbol{X}_i}d_i^{1/2}\kappa_f^{-1/2}$, then gives the result in (6.3.10).

C.3 Dealing with zero examples

As discussed in the main text to simulate the mismatch cavity equations using population dynamics one has to deal with the badly defined zero example case in both (6.4.2) and (6.4.7). One can deal with these factors using the Woodbury identity. The key to this is

realising that for non-zero examples the matrix defined on the right hand side of (6.4.3) is invertible. If we define $\boldsymbol{M}_{d-1}^{\gamma_j+1}$ to be the cavity update (6.4.2) for γ_j+1 excluding the singular $\boldsymbol{h}_i^2$ term then we see that the inverse we require will be given by [48]

$$\begin{aligned}(\boldsymbol{M}_{d-1}^{\gamma_j})^{-1} &= \left(\boldsymbol{M}_{d-1}^{\gamma_j+1} - \boldsymbol{U}\boldsymbol{\Lambda}^{-1}\boldsymbol{U}^{\mathrm{T}}\right)^{-1} \\ &= (\boldsymbol{M}_{d-1}^{\gamma_j+1})^{-1} - (\boldsymbol{M}_{d-1}^{\gamma_j+1})^{-1}\boldsymbol{U}\left(\boldsymbol{\Lambda} + \boldsymbol{U}^{\mathrm{T}}(\boldsymbol{M}_{d-1}^{\gamma_j+1})^{-1}\boldsymbol{U}\right)^{-1}\boldsymbol{U}^{\mathrm{T}}(\boldsymbol{M}_{d-1}^{\gamma_j+1})^{-1} \end{aligned} \tag{C.3.1}$$

with

$$\boldsymbol{\Lambda} = \begin{pmatrix} -\sigma_f^2 - \sigma_g^2 & 0 \\ 0 & \frac{\gamma_j(\sigma_f^2+\sigma_g^2)}{\sigma_f^4} \end{pmatrix} \tag{C.3.2}$$

and $\boldsymbol{U}$ a $(4p_f + 2p_g + 5) \times 2$ matrix with entries

$$U_{\mu\nu} = \delta_{\mu,1}\delta_{\nu,1} - \delta_{\mu,2p_f+3}\delta_{\nu,1} + \delta_{\mu,2p_f+2p_g+5}\delta_{\nu,2} \tag{C.3.3}$$

We find numerically that after the transformation above all remaining inverses can be calculated.

REFERENCES

[1] Reka A. and A. Barabasi. Statistical mechanics of complex networks. *Reviews of Modern Physics*, 74(1):47–97, 2002.

[2] R. Abou-Chacra, D. J. Thouless, and P. W. Anderson. A selfconsistent theory of localization. *J Phys C Solid State*, 6(10):1734–1752, 1973.

[3] W. Aiello, F. Chung, and L. Lu. A random graph model for massive graphs. In *Proceedings of the thirty-second annual ACM symposium on Theory of computing*, pages 171–180, New York, NY, USA, 2000. ACM.

[4] R. Albert, H. Jeong, and A. Barabasi. Diameter of the world wide web. *Nature*, 401(6749):130–131, 1999.

[5] S. Amari, N. Fujita, and S. Shinomoto. Four types of learning curves. *Neural Comput.*, 4(4):605–618, 1992.

[6] S. Amari, N. Murata, K.-R. Muller, M. Finke, and H.H. Yang. Asymptotic statistical theory of overtraining and cross-validation. *Neural Networks, IEEE Transactions on*, 8(5):985 –996, 1997.

[7] A. Annibale and A. C. C. Coolen. What you see is not what you get: how sampling affects macroscopic features of biological networks. *Interface Focus*, 1(6):836–856, 2011.

[8] N. Aronszajn. Theory of reproducing kernels. *T Am Math Soc*, 68:337–404, 1950.

[9] A. L. Barabasi and R. Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, 1999.

[10] D. Barber. *Bayesian Reasoning and Machine Learning.* Cambridge University Press, Cambridge, UK, 2012.

[11] M. Belkin and P. Niyogi. Using manifold structure for partially labeled classification. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in neural information processing systems*, volume 15, pages 929–936, Cambridge, MA, USA, 2003. The MIT Press.

[12] M. Belkin, I. Matveeva, and P. Niyogi. Regularization and semi-supervised learning on large graphs. In J. Shawe-Taylor and Y. Singer, editors, *Learning Theory, Proceedings*, volume 3120, pages 624–638, Berlin, Germany, 2004. Springer-Verlag Berlin.

[13] L. Berg. Asymptotics of integrals, series, and operators. In R. Wong, editor, *Asymptotics and computational analysis*, volume 124 of *Lecture notes in pure and applied mathematics*, pages 35–52, New York, NY, USA, 1990. Marcel Dekker Inc.

[14] P. Berkhin. A survey on PageRank computing. *Internet mathematics*, 2(1):73–120, 2005.

[15] C. Berrou, A. Glavieux, and P. Thitimajshima. Near shannon limit error – correcting coding and decoding – turbocodes. In *Proceedings of the IEEE International Conference on Communications*, volume 1–3, pages 1064–1070, New Yory, NY, USA, 1993. IEEE.

[16] H. A. Bethe. Statistical theory of superlattices. In *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences*, volume 150, pages 552–575, London, UK, 1935. The Royal Society.

[17] S. Bhamidi, R. van der Hofstad, and G. Hooghiemstra. First passage percolation on random graphs with finite mean degrees. *Annals of Applied Probability*, 20(5): 1907–1965, 2010.

[18] G. Bianconi, A. C. C. Coolen, and C. J. P. Vicente. Entropies of complex networks with hierarchically constrained topologies. *Phys Rev E*, 78(1):016114, 2008.

[19] M. Biehl and H. Schwarze. Learning by online gradient descent. *Journal of Physics A - Mathematical and General*, 28(3):643–656, 1995.

[20] C Bishop. *Pattern Recognition and Machine Learning*. Springer, New York, NY, USA., 2nd edition, 2007.

[21] T. S. Blyth and E. F. Robertson. *Basic Linear Algebra*. Springer undergraduate mathematics series. Springer, London, UK, 2002.

[22] B. Bollobás. *Random Graphs*, volume 73 of *Cambridge Studies in Advanced Mathematics*. Cambridge University Press, Cambridge, UK, 2 edition, 2001.

[23] T. Britton, M. Deijfen, and A. Martin-Loeff. Generating simple random graphs with prescribed degree distribution. *J Stat Phys*, 124(6):1377–1397, 2006.

[24] O. Chapelle, J. Weston, and B. Schölkopf. Cluster kernels for semi-supervised learning. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems*, volume 15, pages 489–496, Cambridge, MA, USA, 2003. The MIT Press.

[25] F. K. Chung. *Spectral graph theory*, volume 92 of *Regional conference series in mathematics*. American Mathematical Society, Providence, RI, USA, 1996.

[26] F. K. Chung and L. Lu. *Complex graphs and networks*, volume 107 of *Regional Conference series in mathematics*. American Mathematical Society, Providence, RI, USA, 2006.

[27] F. K. Chung and S. T. Yau. Coverings, heat kernels and spanning trees. *Electron J Comb*, 6:R12, 1999.

[28] F. K. Chung and S. T. Yau. Discrete green's functions. *J Comb Theory A*, 91 (1–2):191–214, 2000.

[29] F. K. Chung, L. Lu, T. Dewey, and Galas D. Duplication models for biological networks. *J Comput Biol*, 10(5):677–687, 2003.

[30] A. C. C. Coolen, R. Kühn, and P. Sollich. *Theory of neural information processing systems.* Oxford University Press, London, UK, 2005.

[31] A. C. C. Coolen, A. De Martino, and A. Annibale. Constrained markovian dynamics of random graphs. *J Stat Phys*, 136(6):1035–1067, 2009.

[32] E. T. Copson. *Theory of functions of a complex variable.* Oxford University Press, London, UK, 1950.

[33] O. Costin. *Asymptotics and Borel Summability*, volume 141 of *Monographs and surveys in pure and applied mathematics.* Chapman and Hall/CRC, Boca Raton, FL, USA, 2008.

[34] J. C. Delvenne and A. S. Libert. Centrality measures and thermodynamic formalism for complex networks. *Phys Rev E*, 83(4):046117, 2011.

[35] L. Devroye. *Non-Uniform Random Variate Generation.* Springer-Verlag, New York, NY, USA, 1986.

[36] H. Drucker, C. J. C. Burges, L. Kaufman, A. Smola, and V. Vapnik. Support vector regression machines. In M. C. Mozer, M. I. Jordan, and T. Petsche, editors, *Advances in Neural Information Processing Systems*, volume 9, pages 155–161, Cambridge, MA, USA, 1997. The MIT Press.

[37] S. F. Edwards and P. W. Anderson. Theory of spin glasses. *J Phys F Met Phys*, 5(5):965–974, 1975.

[38] R. B. Eggleton and D. A. Holton. Simple and multigraphic realizations of degree sequences. In K. L. McAvaney, editor, *Combinatorial Mathematics VIII*, volume 884 of *Lecture Notes in Mathematics*, pages 155–172, Berlin, Germany, 1981. Springer-Berlin.

[39] P. Erdős and A. Rényi. On random graphs, I. *Publicationes Mathematicae (Debrecen)*, 6:290–297, 1959.

[40] P. Erdős and A. Rényi. On the evolution of random graphs. *B Int Statist Inst*, 38 (4):343–347, 1960.

[41] M. V. Fedoryuk. *Metod perevala.* Izdat, Moscow, Russia, 1977.

[42] P. Feynman and A. Hibbs. *Quantum Mechanics and Path Integrals*. Dover Publications, Mineola, NY, USA, 2010.

[43] F. Font-Clos, F. A. Massucci, and I. P. Castillo. A weighted belief-propagation algorithm to estimate volume-related properties of random polytopes. *J Stat Mech Theory E*, 2012(11):P11003, 2012.

[44] J. Freeman and D. Saad. Dynamics of on-line learning in radial basis networks. *Phys Rev E*, 56(1):907–918, 1997.

[45] R. G. Gallager. *Low-density parity-check codes*. The MIT Press, Cambridge, MA, USA, 1963.

[46] M. G. Genton. Classes of kernels for machine learning: A statistics perspective. *J Mach Learn Res*, 2(2):299–312, 2002.

[47] C. Goffman. And what is your Erdős number? *The American Mathematical Monthly*, 76(7):791, 1969.

[48] W.W. Hager. Updating the inverse of a matrix. *SIAM Review*, 31(2):221–239, 1989.

[49] D. Haussler, M. Kearns, H. S. Seung, and N. Tishby. Rigorous learning curve bounds from statistical mechanics. *Mach Learn*, 25(2–3):195–236, 1996.

[50] M. Herbster and G. Lever. Predicting the labelling of a graph via minimum p-seminorm interpolation. In *The 22nd Annual Conference on Learning Theory (COLT 2009)*, Red Hook, NY, USA, 2009. Curran Associates Inc.

[51] M. Herbster and M. Pontil. Prediction on a graph with a perceptron. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in neural information processing systems 19*, pages 577–584, Cambridge, MA, USA, 2007. The MIT Press.

[52] M. Herbster, M. Pontil, and L. Wainer. Online learning over graphs. In *Proceedings of the 22nd international conference on Machine learning*, pages 305–312, New York, NY, USA, 2005. ACM.

[53] M. Herbster, G. Lever, and M. Pontil. Online prediction on large diameter graphs. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in neural information processing systems*, volume 21, pages 649–656, Red Hook, NY, USA, 2009. Curran Associates Inc.

[54] E. J. Hinch. *Perturbation Methods.* Cambridge University Press, Cambridge, UK, 1991.

[55] C. E. Hutchinson. The Kalman filter applied to aerospace and electronic systems. *IEEE transactions on aerospace and electronic systems*, 20(4):500–504, 1984.

[56] S. Janson. Asymptotic equivalence and contiguity of some random graphs. *Random Structures & Algorithms*, 36(1):26–45, 2010.

[57] H. Jeong, B. Tombor, R. Albert, Z. Oltvati, and A. Barabasi. The large scale organization of metabolic networks. *Nature*, 407(6804):651–654, 2000.

[58] M. H. Kalos and P. A. Whitlock. *Monte Carlo Methods.* John Wiley and Sons, Darmstadt, Germany, 2009.

[59] D. Kaminski. On the n-variable saddle point and steepest descent methods. In R. Wong, editor, *Asymptotic and computational analysis*, volume 124 of *Lecture notes in pure and applied mathematics*, pages 627–637, New York, NY, USA., 1990. Marcel Dekker Inc.

[60] G. Kimeldorf and G. Wahba. A correspondence between Bayesian estimation on stochastic processes and smoothing by splines. *Annals of mathematical statistics*, 41(2):495–502, 1970.

[61] R. Kindermann and J. Snell. *Markov random fields and their applications.* American Mathematical Society, Providence, RI, USA, 1980.

[62] R. Kondor and J. Lafferty. Diffusion kernels on graphs and other discrete structures. In C. Sammut and Hoffmann A. G., editors, *Proceedings of the 19th International Conference on Machine Learning (ICML)*, pages 315–322, San Francisco, CA, USA., 2002. Morgan Kauffmann.

[63] A. Kovac and A.D.A.C. Smith. Nonparametric regression on a graph. *J Comput Graph Stat*, 20(2):432–447, 2011.

[64] R. Kühn. Spectra of sparse random matrices. *J Phys A Math Gen*, 41(29):295002, 2008.

[65] R. Kühn and J. van Mourik. Spectra of modular and small-world matrices. *J Phys A Math Gen*, 44(16):165205, 2011.

[66] R. Kühn, J. van Mourik, M. Weigt, and A. Zippelius. Finitely coordinated models for low-temperature phases of amorphous systems. *J Phys A Math Gen*, 40(31): 9227–9252, 2007.

[67] S. Kullback. The Kullback-Leibler distance. *The American Statistician*, 41(4): 340–341, 1987.

[68] J. Langford. Tutorial on practical prediction theory for classification. *J Mach Learn Res*, 6:273–306, 2005.

[69] N. Linial, Y. Mansour, and R. L. Rivest. Results on learnability and the Vapnik-Chervonenkis dimension. *Information and Computation*, 90(1):33–49, 1991.

[70] L. Lovàsz. Random walks on graphs: A survey. In D. Miklos, V. T. Sos, and T. Szonyi, editors, *Combinatorics, Paul Erdős is eighty*, volume 2 of *Bolyai society mathematical studies*, pages 353–397, Budapest, Hungary, 1993. Bolyai Janos Matematika Tarsulat.

[71] D. Malzahn and M. Opper. Learning curves and bootstrap estimates for inference with Gaussian processes: A statistical mechanics study. *Complexity*, 8(4):57–63, 2003.

[72] D. Malzahn and M. Opper. A statistical physics approach for the analysis of machine learning algorithms on real data. *J Stat Mech Theory E*, 2005(11):P11001, 2005.

[73] G. Matheron. The intrinsic random functions and their applications. *Advances in applied probability*, 5(3):439–468, 1973.

[74] D. McAllester. PAC-bayesian model averaging. In *COLT '99 Proceedings of the twelth annual conference on computational learning theory*, pages 164–170, 1999.

[75] B. D. McKay. The expected eigenvalue distribution of a large regular graph. *Linear Algebra and its Applications*, 40:203–216, 1981.

[76] M. Mézard and A. Montanari. *Information, Physics, and computation.* Oxford University Press, UK, Oxford, UK, 2009.

[77] M. Mézard and G. Parisi. A replica analysis of the travelling salesman problem. *Journal de Physique*, 47(8):1285–1296, 1986.

[78] M. Mézard and G. Parisi. The Bethe lattice spin glass revisited. *Eur Phys J B*, 20(2):217–233, 2001.

[79] C. Monthus and C. Texier. Random walk on the Bethe lattice and hyperbolic Brownian motion. *J Phys A Math Gen*, 29:2399–2409, 1996.

[80] R. M. Neal. *Bayesian learning for neural networks*, volume 118 of *Lecture notes in statistics.* Springer-Verlag, Berlin, Germany., 1996.

[81] M. Newman. Models of the small world. *J Stat Phys*, 101(3–4):819–841, 2000.

[82] M. Newman and D. J. Watts. Renormalization group analysis of the small-world network model. *Physics Letters A*, 263(4–6):341–346, 1999.

[83] M. Newman, S. Strogatz, and D. Watts. Random graphs with arbitrary degree distributions and their applications. *Phys Rev E*, 64(2):026118, 2001.

[84] I. Norros and H. Reittu. On a conditionally Poissonian graph process. *Adv Appl Probab*, 38(1):59–75, 2006.

[85] J. K. Ochab. Maximal-entropy random walk unifies centrality measures. arxiv preprint arxiv:1206.4094, 2012.

[86] L. Onsager. Electric moments of molecules in liquids. *J Am Chem Soc*, 58:1486–1493, 1936.

[87] M. Opper and D. Haussler. Bounds for predictive errors in the statistical mechanics of supervised learning. *Phys Rev Lett*, 75(20):3772–3775, 1995.

[88] M. Opper and F. Vivarelli. General bounds on Bayes errors for regression with Gaussian processes. In M. S. Kearns, S. A. Solla, and D A Cohn, editors, *Advances in Neural Information Processing Systems*, volume 11, pages 302–308, Cambridge, MA, USA., 1999. The MIT Press.

[89] G. Parisi. A sequence of approximated solutions to the S–K model for spin–glasses. *J Phys A Math Gen*, 13(4):L115–L121, 1980.

[90] G. Parisi. The order parameter for spin-glasses: Function on the interval 0–1. *J Phys A Math Gen*, 13(3):1101–1112, 1980.

[91] G. Parisi. Magnetic properties of spin glasses in a new mean field theory. *J Phys A Math Gen*, 13(5):1887–1895, 1980.

[92] J. Park and M. E. J. Newman. Statistical mechanics of networks. *Physical Review E*, 70(6):066117, 2004.

[93] J. Pearl. *Probabilistic reasoning in intelligent systems: Networks of plausible inference*. Morgan Kauffmann, San Francisco, CA, USA, 1988.

[94] C. J. Pérez-Vicente and A. C. C. Coolen. Spin models on random graphs with controlled topologies beyond degree constraints. *J Phys A Math Gen*, 42(16): 169801, 2009.

[95] T. Poggio and F. Girosi. Networks for approximation and learning. In *Proceedings of the IEEE*, volume 78, pages 1481–1479, New York, NY, USA, 1990. IEEE-Institute of electrical engineers Inc.

[96] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, Cambridge, MA, USA, 2005.

[97] T. Rogers, K. Takeda, I. Castillo, and R. Kühn. Cavity approach to the spectral density of sparse symmetric random matricies. *Phys Rev E*, 78(3):31116–31121, 2008.

[98] T. Rogers, C. Vicente, K. Takeda, and I. Castillo. Spectral density of random graphs with topological constraints. *J Phys A Math Gen*, 43(19):195002, 2010.

[99] J. F. Rual, K. Venkatesan, T. Hao, T. Hirozane-Kishikawa, A. Dricot, N. Li, G. F. Berriz, F. D. Gibbons, M. Dreze, N. Ayivi-Guedehoussou, N. Klitgord, C. Simon, M. Boxem, S. Milstein, J. Rosenberg, D. S. Goldberg, L. V. Zhang, S. L. Wong, G. Franklin, S. M. Li, J. S. Albala, J. H. Lim, C. Fraughton, E. Llamosas, S. Cevik, C. Bex, P. Lamesch, R. S. Sikorski, J. Vandenhaute, H. Y. Zoghbi, A. Smolyar, S. Bosak, R. Sequerra, L. Doucette-Stamm, M. E. Cusick, D. E. Hill, F. P. Roth, and M. Vidal. Towards a proteome-scale map of the human protein-proteion interaction network. *Nature*, 437(7062):1173–1178, 2005.

[100] M. Seeger. PAC-bayesian generalisation error bounds for gaussian process classification. *J Mach Learn Res*, 3:233–269, 2002.

[101] H. S. Seung, H. Sompolinsky, and N. Tishby. Statistical mechanics of learning from examples. *Phys Rev A*, 45(8):6056–6091, 1992.

[102] G. Shakhnarovich, T. Darrell, and P. Indyk. *Nearest-Neighbour Methods in Learning and Vision: Theory and Practice*. The MIT Press, Cambridge, MA, USA, 2006.

[103] J. Shawe-Taylor. PAC-bayes analysis for gaussian process models. Private Communication, 2011.

[104] J. Shawe-Taylor and N. Cristianini. *Kernel methods for pattern analysis*. Cambridge University Press, Cambridge, UK, 2004.

[105] J. Shawe-Taylor, P. Bartlett, R. Williamson, and M. Anthony. A framework for structural risk minimisation. In *COLT '96 Proceedings of the ninth annual conference on Computational learning theory*, pages 68–76, 1996.

[106] D. Sherrington and S. Kirkpatrick. Solvable model of a spin-glass. *Phys Rev Lett*, 35(26):1792–1796, 1975.

[107] N. Shibata, Y. Kajikawa, and K. Matsushima. Topological analysis of citation networks to discover the future core articles. *Journal of the American society for information science and technology*, 58(6):872–882, 2007.

[108] R. Sinatra, J. Gomez-Gardenes, R. Lambiotte, V. Nicosia, and V. Latora. Maximal-entropy random walks in complex networks with limited information. *Phys Rev E*, 83(3):030103, 2011.

[109] A. Smola and R. Kondor. Kernels and regularization on graphs. In B Schölkopf and M. K. Warmuth, editors, *Learning theory and kernel machines*, volume 2777 of *Lecture Notes in Artificial Intelligence*, pages 144–158, Berlin, Germany., 2003. Springer-Verlag Berlin.

[110] A. Smola and B. Schölkopf. From regularization operators to support vector kernels. In M. I. Jordan, M. S. Kearns, and S. A. Solla, editors, *Advances in Neural Information Processing Systems*, volume 10, pages 343–349, Cambridge, MA, USA, 1998. The MIT Press.

[111] P. Sollich. Learning curves for Gaussian processes. In M. S. Kearns, S. A. Solla, and D A Cohn, editors, *Advances in Neural Information Processing Systems*, volume 11, pages 344–350, Cambridge, MA, USA., 1999. The MIT Press.

[112] P. Sollich. Approximate learning curves for Gaussian processes. In *Ninth international conference on artificial neural networks (ICANN99)*, volume 2, pages 437–442, Edison, NJ, USA, 1999. Institute of Electrical Engineers Inspec inc.

[113] P. Sollich. Gaussian process regression with mismatched models. In S Becker T G Dietterich and Z Ghahramani, editors, *Advances in Neural Information Processing Systems*, volume 14, pages 519–526, Cambridge, MA, USA, 2002. The MIT Press.

[114] P. Sollich. Can Gaussian process regression be made robust against model mismatch? In N Lawrence J Winkler and M Niranjan, editors, *Deterministic and Statistical Methods in Machine Learning*, pages 211–228, Berlin, Germany, 2005. Springer-Berlin.

[115] P. Sollich and A. Halees. Learning curves for Gaussian process regression: Approximations and bounds. *Neural Computation*, 14(6):1393–1428, 2002.

[116] P. Sollich and C. K. I. Williams. Using the equivalent kernel to understand Gaussian process regression. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in*

Neural Information Processing Systems, volume 17, pages 1313–1320, Cambridge, MA, USA, 2005. The MIT Press.

[117] A. Srivastava and M. Sahami, editors. *Text Mining: Classification, Clustering and Applications*. Chapman and Hall/CRC, Boca Raton, FL, USA, 2009.

[118] A. Steger and N. Wormald. Generating random regular graphs quickly. *Combinatorics, Probability and Computing*, 8:377, 1999.

[119] J. Stelling. Metabolic network structure determines key aspects of functionality and regulation. *Nature*, 420(6912):190–193, 2002.

[120] M. Szummer and L. Jaakkola. Partially labelled classification with Markov random walks. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems*, volume 14, Cambridge, MA, USA, 2002. The MIT Press.

[121] R. Taylor. Constrained switchings in graphs. In K. L. McAvaney, editor, *Combinatorial Mathematics VIII*, volume 884 of *Lecture Notes in Mathematics*, pages 314–336, Berlin, Germany, 1981. Springer-Berlin.

[122] D. J. Thouless, P. W. Anderson, and R. G. Palmer. Solution of 'solvable model of a spin glass'. *Philosophical magazine*, 35(3):593–601, 1977.

[123] V. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, New York, NY, USA, 1998.

[124] V. Vapnik and A. Chervonenkis. Ordered risk minimization I. *Automation and remote control*, 35(8):1226–1235, 1974.

[125] V. Vapnik and A. Chervonenkis. Ordered risk minimization II. *Automation and remote control*, 35(9):1403–1412, 1974.

[126] L. Viana and A. Bray. Phase-diagrams for dilute spin-glasses. *J Phys C Solid State*, 18(15):3037–3051, 1985.

[127] F. Viger and M. Latapy. Efficient and simple generation of random simple connected graphs with prescribed degree sequence. In *Computing and combinatorics,*

proceedings, volume 3595 of *Lecture notes in computer science*, pages 440–449, Berlin, Germany, 2005. Springer-Verlag Berlin.

[128] G. Wahba. *Spline Models for Observational data*. CBMS-NSF Regional Conference Series in Applied Mathematics. SIAM, Philadelphia, PA, USA, 1990.

[129] A. Wald. Contributions to the theory of statistical estimation and testing hypotheses. *Annals of Mathematical Statistics*, 10:299–326, 1939.

[130] T. L. H. Watkin, A. Rau, and M. Biehl. The statistical mechanics of learning a rule. *Rev Mod Phys*, 65(2):499–556, 1993.

[131] D. Watts and S. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393(6684):440–442, 1998.

[132] C .K. I. Williams. Prediction with Gaussian processes: From linear regression to linear prediction and beyond. In M. Jordan, editor, *Learning in Graphical Models*, pages 599–618. Kluwer Academic Publishers, Dordecht, The Netherlands, 1998.

[133] C. K. I. Williams and F. Vivarelli. Upper and lower bounds on the learning curve for Gaussian processes. *Mach Learn*, 40:77–102, 2000.

[134] R. Witte. *Statistics*. Holt, Rinehart and Winston, Orlando, FL. USA, 1989.

[135] E. Wong. *Stochastic Processes in Information and Dynamical Systems*. McGraw-Hill, New York, NY, USA, 1971.

[136] J. S. Yedidia, W. T. Freeman, and Y. Weiss. *Exploring Artificial intelligence in the new millenium*, chapter 8, pages 239–269. Morgan Kauffmann, San Francisco, CA, USA, 2002.

[137] A. L Yuille and N. M. Grzywacz. A mathematical analysis of the motion coherence theory. *International journal of computer vision*, 3(2):155–175, 1989.

[138] M. Zhongwei, L. Zongxiang, and S. Jingyan. Comparison analysis of the small-world topological model of chinese and american power grids. *Automation of electric power systems*, 28(15):21–24, 2004.

[139] X. Zhu, Z. Ghahramani, and J. Lafferty. Semi-supervised learning using Gaussian fields and harmonic functions. In *20th international conference on machine learning*, pages 912–919, Palo Alto, CA, USA, 2003. AAAI Press.